

HIGHCHARTS

目錄

Highcharts中文教程

> 来源：Highcharts中文教程

中文教程开始陆陆续续更新啦，本次由本人独立完成，参考官方教程及根据个人经验编写，由于工作较忙，基本上是晚上抽空完成，某些地方思路模糊或者语言表述不清楚还请指正。

2015-09-06，夜，杭州

（正文完，最后更新时间：2015-09-06 23:43:24）

Highcharts入门（100%）

> Highcharts入门

章节进度

已完成 100%，最后更新时间：2014-01-30

章节目录

（正文完，最后更新时间：2014-07-23 15:52:27）

Highcharts简介

> Highcharts:功能强大、开源、美观、图表丰富、兼容绝大多数浏览器的纯js图表库

```
Highcharts是一款纯javascript编写的图表库，能够很简单便捷的在Web网站或Web应用中添加交互性的图表，Highc
```

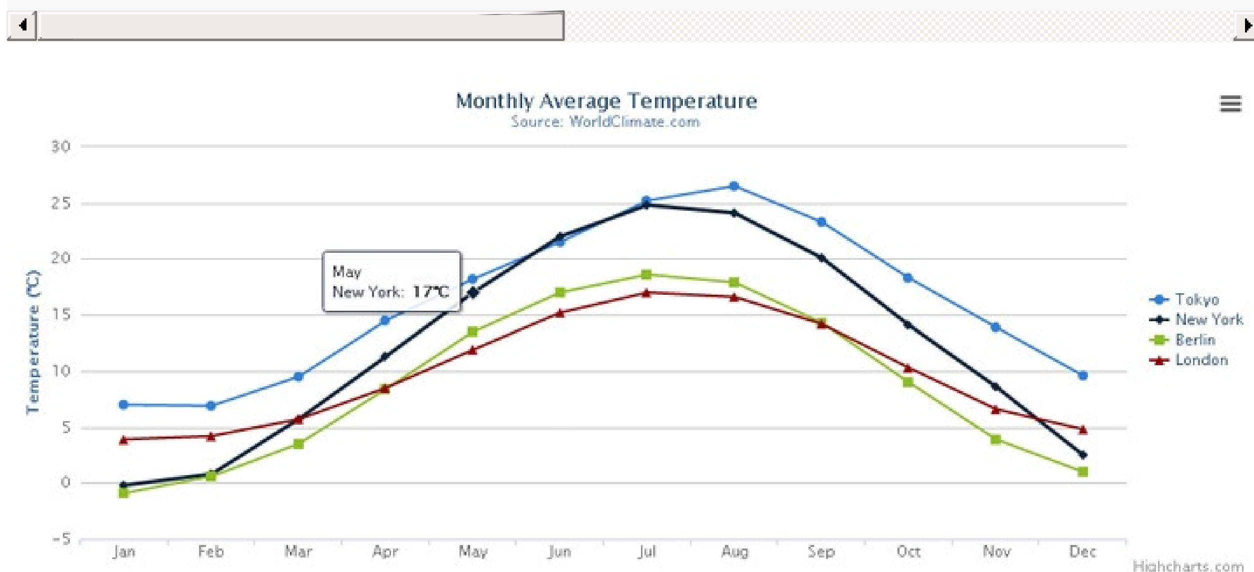


图**1-1 Highcharts Basic Chart**

由于其功能强大、简单易用、开源免费等优点，Highcharts在国内外越来越受欢迎。下面详细说明Highcharts的优势

Highcharts优势

兼容性

Highcharts支持目前所有的现代浏览器，包括IE6 +、iPhone/iPad、Android。Highcharts在标准（W3C标准）浏览器中使用SVG技术渲染图形，在遗留的IE浏览器中使用VML技术来绘图。

开源免费

针对个人用户及非商业用途免费，并提供源代码下载，你可以任意的修改它。商业用途需要购买许可，个人及非商业用途须遵循CC BY-NC 3.0协议

纯Javascript

Highcharts完全基于本地浏览器技术，不需要任何插件（例如Flash、java），不需要安装任何服务器环境或动态语言库支持，只需要两个js文件即可运行。

图表类型丰富

Highcharts目前支持直线图、曲线图、面积图、曲线面积图、面积范围图、曲线面积范围图、柱状图、柱状范围图、条形图、饼图、散点图、箱线图、气泡图、误差线图、漏斗图、仪表图、瀑布图、雷达图，共18种类型图表，其中很多图表可以集成在同一个图形中形成综合图。

动态性

提供丰富的API接口，方便在创建图表后对图表的任意点、线和文字等进行增加、删除和修改操作。支持众多的Javascript事件，结合jQuery、MooTools、Prototype等javascript框架提供的Ajax接口，可以实时地从服务器取得数据并实时刷新图表。

多轴支持

对于需要比较的数据，Highcharts提供多轴支持。并且可以针对每个轴设置其位置、文字和样式等属性。

动态提示框

当鼠标悬停在图表上的数据点时，Highcharts会显示信息提示框，当然，显示的内容和样式可以自己指定和设置。

图表导出和打印功能

你可以将Highcharts图表导出为PNG、JPG、PDF和SVG格式文件或直接在网页上打印出来。

图表缩放

可以设置图表的缩放，让你更方便查看图表数据。

支持外部数据加载

Highcharts支持多种数据形式，可以是Javascript数组、json文件、json对象和表格数据等，这些数据来源可以是本地、不同页面，甚至是不同网站。

（正文完，最后更新时间：2014-07-23 15:53:39）

Highcharts下载与使用

> 下载即可运行，看例子代码就可以入门

Highcharts 所有的源代码及例子都可以通过官网、中文网下载得到。

下载

- 中文网下载中心：http://www.hcharts.cn/product/download.php
- 官方下载：http://www.highcharts.com/download
- Github 开源项目：https://github.com/highslide-software/highcharts.com
- 中文网开放CDN ： http://cdn.hcharts.cn
- 官方 CDN 服务 ： http://code.highcharts.com

使用

解压下载得到的压缩包，进入相应的目录查看所有包含文件，Highcharts 提供的文件目录如下图所示：



目录结构说明：

```
|-- examples           例子目录
|-- exporting-server   导出服务器目录
|-- gfx                图片资源目录
|-- graphics           图片资源目录
|-- js                 所有 js 文件源码代码（带 .src 的文件为未压缩版源代码）
|-- index.htm          例子入口文件
```

建议初学者从官方提供的例子代码入手，Highcharts 提供的丰富例子可以让你轻松入门。

注意

由于官方提供的例子中用到的 jquery.js 是引用 google 的服务（http://ajax.googleapis.com/ajax/libs/jquery/1.8.2/jquery.min.js），由于众所周知的原因，这个服务在国内访问并不稳定（甚至无法是使用），导致打开相应的例子后显示空白，请参考"关于从官网上下载Highcharts后运行例子空白的说明" 或直接访问 "在线实例"

在线实例

- Highcharts 在线演示
- Highstock 在线演示
- Highmaps 在线演示

开放CDN

如果你想直接引用在线资源，建议使用中文网提供的开放CDN服务。

开放**CDN**服务

（正文完，最后更新时间：2015-05-27 10:45:36）

Highcharts配置

> 只需要引用两个 JS 文件即可运行

Highcharts 的运行需要两个 JS 文件， `highcharts.js` 及 jQuery 、 MooTools 、Prototype 、Highcharts Standalone Framework 常用 JS 框架中的一个。

引入 JS 文件可以是引入本地文件和在线文件，针对不同的 JS 框架需要引入的文件有所不同，下面详细说明。

一、引入在线资源

1、jQuery

jQuery 是目前使用最广泛的 JS 框架，无特殊说明，本教程所用的环境及所有例子都是基于 jQuery 的。

```
<script src="http://cdn.hcharts.cn/jquery/jquery-1.8.3.min.js"></script>
<script src="http://cdn.hcharts.cn/highcharts/highcharts.js"></script>
```

2、Highcharts Standalone Framework

jQuery 目前用的最广泛，但是其体积过大（100K 以上），在移动端网络带宽有限的情况下，并不是最优选择，如果你的页面没有其他地方需要用到 jQuery，jQuery 仅仅用于 highcharts，这种情况可以考虑使用 `Highcharts Standalone Framework` ，Highcharts Standalone Framework 压缩后的大小只有 `2k` 哦。

```
<script src="http://cdn.hcharts.cn/highcharts/adapters/standalone-framework.js"></script>
<script src="http://cdn.hcharts.cn/highcharts/highcharts.js" ></script>
```

关于使用 Highcharts Standalone Framework 的实例及注意事项见：http://highcharts.me/article/19

3、MooTool 或 Prototype

使用 MooTools 或 Prototype 需要额外的引入 Highcharts 提供的适配器。 使用 MooTools 需要引入的文件如下：

```
<script src="http://cdn.hcharts.cn/mootools/MooTools-Core-1.5.1.js"></script>
<script src="http://code.highcharts.com/adapters/mootools-adapter.js"></script>
<script src="http://cdn.hcharts.cn/highcharts/highcharts.js"></script>
```

使用 Prototype 需要引入的文件如下：

```
<script src="http://cdn.hcharts.cn/prototype/prototype-1.7.2.js"></script>
<script src="http://code.highcharts.com/adapters/prototype-adapter.js"></script>
<script src="http://cdn.hcharts.cn/highcharts/highcharts.js"></script>
```

二、引入本地文件

上节说到 Highcharts 所有 JS 文件都可以通过下载获得，如果你不想引入在线资源，可以直接引入本地文件。

本地引入文件同在线引入，这里拿 jQuery 来举例说明

```
<script src="js/jquery.min.js"></script>
<script src="js/highcharts.js"></script>
```

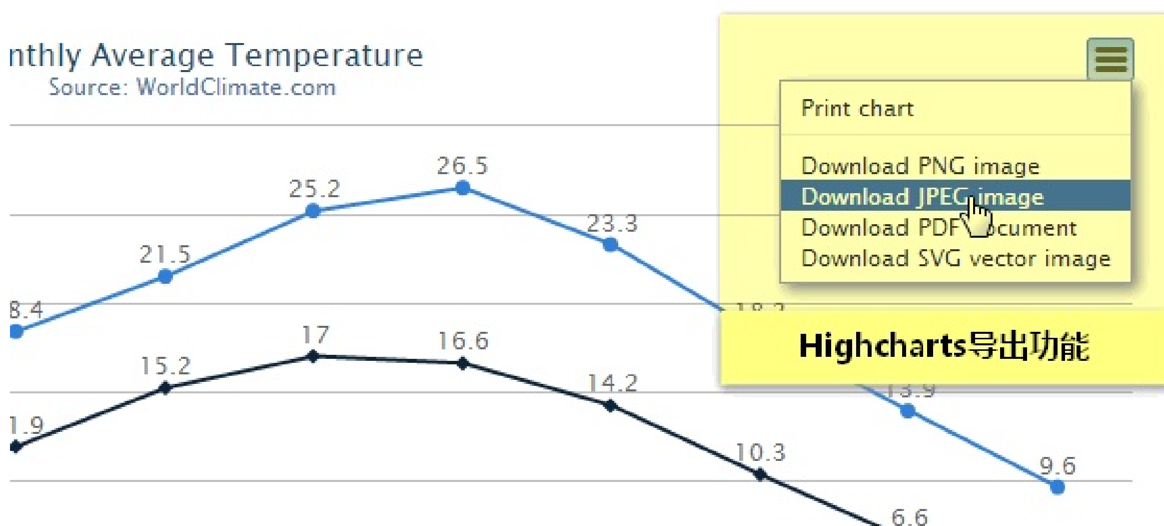
三、高级功能

Highcharts 提供了图表导出、图表主题等其他额外功能，这些功能的实现，需要额外引入相关 JS 文件。

1、图表导出功能

导出功能是将图表导出下载为常见图片文件或 PDF 文档的功能，想要使用这个功能，额外引入 `exporting.js` 即可

```
<script src="http://cdn.hcharts.cn/highcharts/modules/exporting.js" type="text/javascript
```



2、图表主题

Highcharts 提供图表更换主题功能，引入想应的主题 JS 文件即可改变图表样式。除默认主题样式外，Highcharts 官方提供多款主图，你也可以根据需要自己设计图表主题。

Highcharts 提供的主题文件放置在 `/js/themes/` 目录下，给图表添加灰色（Gray）主题的代码是：

```
<script src="http://cdn.hcharts.cn/highcharts/themes/gray.js" type="text/javascript"></sc
```

（正文完，最后更新时间：2015-12-04 12:08:04）

Hello World程序

> 本人精通c、c++、java、C#、Ruby、Perl、Lisp、Python、Objective-C、ActionScript、Pascal、JavaScript、PHP、ASP等语言的HelloWorld程序编写

新建一个html文件，将highcharts引入到你的页面后，通过两个步骤我们就可以创建一个简单的图表了。

1、创建**div**容器

在页面的 body部分创建一个div，并指定div 的 id，高度和宽度，代码如下

```
<div id="container" style="min-width:800px;height:400px"></div>
```

2、编写**Highcharts**代码

编写Highcharts必须的代码，用<script></script>包裹，代码可以放在页面的任意地方，一个最简单的图表实例代码如下

```
$(function () {
    $('#container').highcharts({                    //图表展示容器，与div的id保持一致
        chart: {
            type: 'column'                          //指定图表的类型，默认是折线图（line）
        },
        title: {
            text: 'My first Highcharts chart'       //指定图表标题
        },
        xAxis: {
            categories: ['my', 'first', 'chart']   //指定x轴分组
        },
        yAxis: {
            title: {
                text: 'something'                   //指定y轴的标题
            }
        },
        series: [{                                  //指定数据列
            name: 'Jane',                           //数据列名
            data: [1, 0, 4]                         //数据
        }, {
            name: 'John',
            data: [5, 7, 3]
        }]
    });
});
```

完成上述两个步骤后，保存文件并用浏览器打开，运行结果如下图所示

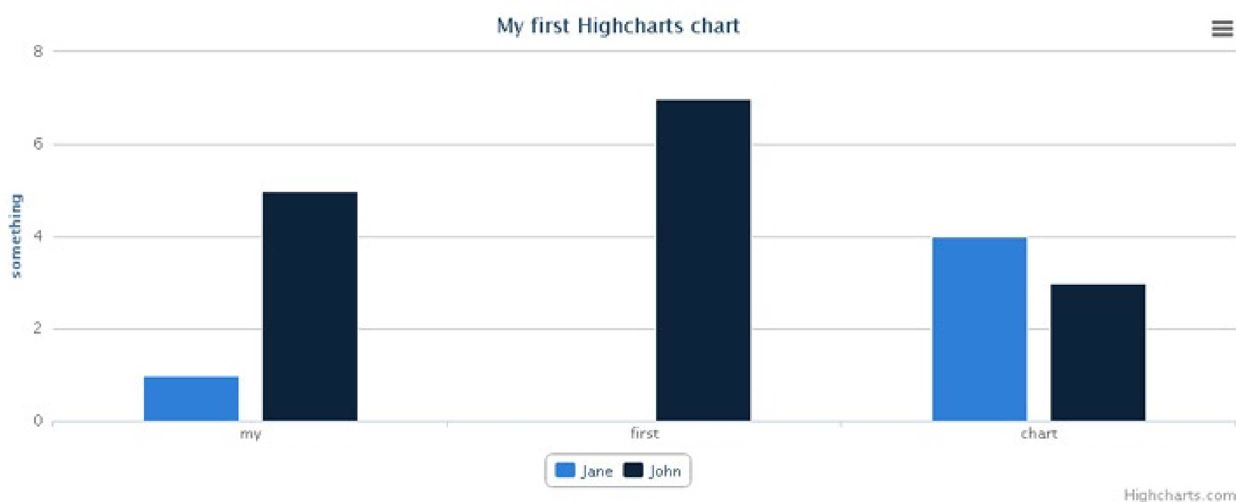


图**1-5 My first chart**

在线试一试

上述例子完整**html**代码如下 **<small>**你可以拷贝保存为**html**文件运行即可看到效果**</small>**

```
<html>
<head>
   <script type="text/javascript" src="http://cdn.hcharts.cn/jquery/jquery-1.8.3.min.js">
   <script type="text/javascript" src="http://cdn.hcharts.cn/highcharts/highcharts.js"></
   <script>
     $(function () {
        $('#container').highcharts({
            chart: {
                type: 'column'
            },
            title: {
                text: 'My first Highcharts chart'
            },
            xAxis: {
                categories: ['my', 'first', 'chart']
            },
            yAxis: {
                title: {
                    text: 'something'
                }
            },
            series: [{
                name: 'Jane',
                data: [1, 0, 4]
            }, {
                name: 'John',
                data: [5, 7, 3]
            }]
        });
    });
   </script>
</head>

<body>
   <div id="container" style="min-width:800px;height:400px;"></div>
</body>
</html>
```

至此，我们已经初步了解和学会了自己编写简单的Highcharts图表了，这只是开始，后面的会越来越有趣。

如何学习Highcharts

Highcharts的配置（或者说Highcharts代码）是一个 `json` 字符串，类似于 `{chart:{type:"spline"}}` ，json具有易于人阅读和编写的特点，所以学习和开发Highcharts并不难，只是熟悉API程度的问题。所以本教程的重点是带大家熟悉API文档加一点点处理技巧，更多的是靠大家花时间学习和实践。

准备知识

- 熟悉Html、jQuery、Json、Ajax等前端知识
- 至少会一种后台语言，例如Php、Java、asp等（Highcharts只是做数据展现，具体的数据来源必须通过动态语言）

几点建议

- 任何东西的掌握必须通过自我实践
- 多看API，Highcharts提供的API文档非常完善，几乎所有的问题都可以在API找到解决办法

（正文完，最后更新时间：2015-05-22 12:49:19）

Highcharts兼容性

> 兼容所有现代浏览器，适配主流js框架

Highcharts完全基于本地浏览器技术，不需要任何插件，甚至不需要安装任何服务器环境，只需要两个js文件即可运行。Highcharts针对不同的js框架jQuery、Mootools、Prototype做了适配工作，不同框架开发人员不需要再额外学习其他框架即可上手。

highcharts可以运行在任何现代浏览器，包括移动终端以及IE6，标准的浏览器用 `SVG` 技术渲染图表，对于遗留的浏览器，则用 `VML` 来绘图。

浏览器兼容性测试

浏览器厂商	浏览器名	支持版本
Microsoft	Internet Explorer	6.0 +
Mozilla	Firefox	2.0 +
Google	Chrome	1.0 +
Apple	Safari	4.0 +
Opera	Opera	9.0 +
Apple	iOS(Safari)	3.0 +
Google	Android Borwser	2.0 + *

说明：对android浏览器部分支持，点击查看详细。

Javascript框架支持

目前已经测试过的框架及版本如下

JQuery	Mootools	Prototype
1.8.2	1.4.5	1.7
1.7.2	1.3.2	
1.6.2	1.2.5	
1.5.2		
1.4.4		
1.3.2		

其他版本都能正常使用只是没有测试。

图表渲染引擎及性能

不同的浏览器支持不同的渲染技术，现在大多数浏览器支持 `SVG`，只有较老的版本的IE不支持，这些较老的浏览器用 `VML` 绘制图表，针对不同浏览器图表绘制性能测试结果如下

浏览器及版本	使用的渲染技术	性能
Internet Explorer 9	SVG	非常快
Internet Explorer 8	VML	快
Internet Explorer 7	VML	慢
Internet Explorer 6	VML	慢
Firefox	SVG	非常快
Chrome	SVG	非常快
Safari	SVG	非常快
Opera	SVG	非常快
iOS Safari	SVG	快
Android 3+	SVG	快
Android 2.x	Canvas	慢

Android 2.x

Android 2.x 没有提供 SVG 支持，针对Android 2.x，Highcharts 开发出基于canvg的独立渲染器，但是其有如下限制：

- 数据提示框数据共享总是启用（即 tooltip.shared = true）
- 图表在第一次渲染时，会从 `code.highcharts.com` 下载 **canvg** 渲染器 **+ rgbcolor.js + canvg.js** （包含在同一个文件里），这个地址可以通过 global.canvasToolsURL 属性指定。
- 图表和数据列（series）动画效果无效（无动画效果）
- 点击图例（legend）无法显示或隐藏数据列（series）
- 数据列（series）和数据点（point）的触摸事件无效
- 图表缩放（zoom）无效
- 使用渲染API直接添加图形到图表上无效

（正文完，最后更新时间：2015-09-11 09:47:09）

Highcharts使用许可

> 开源 != 免费

Highcharts是一款开源图表库，开源但不完全免费。Highcharts针对个人用户及非商业用途免费，商业用途需要购买授权。

1、Free - Non-commercial

针对个人用户及非商业用途免费。

使用时请遵守 CC BY-NC 3.0 (Creative Commons Attribution-NonCommercial 3.0 License)协议。

CC BY-NC 3.0 中文详情请参考：署名-非商业性使用 3.0 中国大陆 (CC BY-NC 3.0 CN)

2、Commercial and government licenses

针对商业用户及政府的商业授权许可，商用License分为三种：网站授权、开发者授权、OEM授权。

Highcharts商用授权费用详见下图：

Highcharts | Highstock | Highmaps | Highslide

序号	License名称	价格（元）	服务费（%）*
# 1	Highcharts Single Website	900	10
# 2	Highcharts Single developer	3900	8
	Maintenance and Support for Highcharts Single developer *	3100	
# 3	Highcharts 5 developers	15000	6
	Maintenance and Support for Highcharts 5 developers *	12000	
# 4	Highcharts 10 developers	25000	4
	Maintenance and Support for 10 Highcharts developers *	20000	
# 5	Highcharts OEM License	邮件咨询	2

更多关于商用授权信息详见：**Licenses**代理 页面

（正文完，最后更新时间：2014-06-24 18:40:52）

Highcharts基础教程（67%）

章节进度

已完成 93%，最后更新时间：2015-09-29

章节目录

（正文完，最后更新时间：2015-09-29 20:01:48）

Highcharts主要组成

通常情况下，Highcharts包含标题（Title）、坐标轴（Axis）、数据列（Series）、数据提示框（Tooltip）、图例（Legend）、版权信息（Credits）等，高级的还包括导出功能按钮（Exporting）、标示线（PlotLines）、标示区域（PlotBands）等。

Highcharts基本组成部分如下图所示

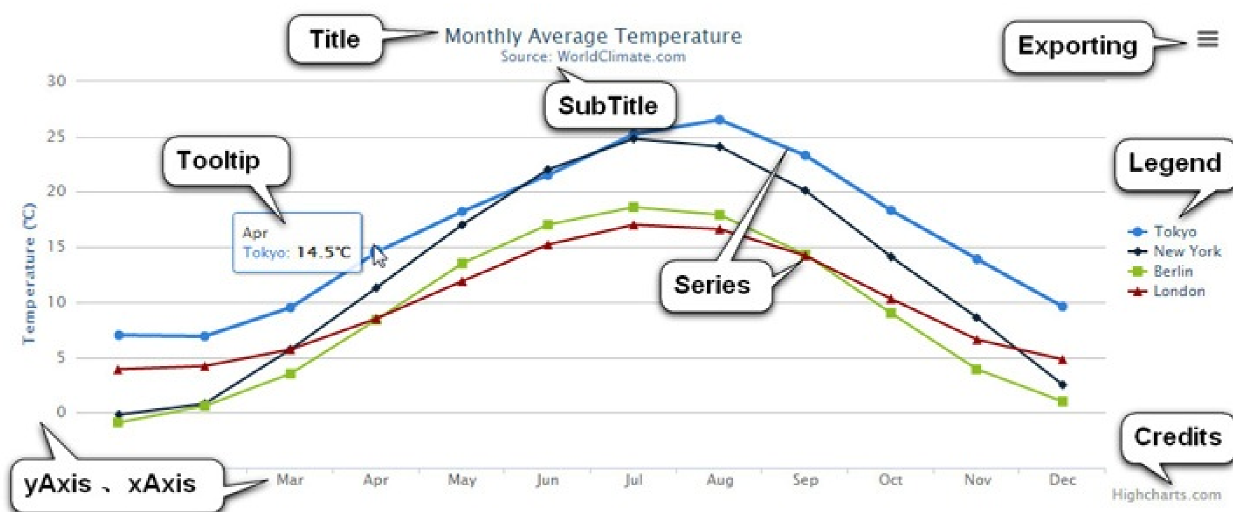


图**2-1 Highcharts**基本组成部分

Highcharts主要组成

Title

图表标题，包含标题和副标题（subTitle），其中副标题是非必须的。

Axis

坐标轴，包含x轴（xAxis）和y轴（yAxis）。通常情况下，x轴显示在图表的底部，y轴显示在图表的左侧。多个数据列可以共同使用同一个坐标轴，为了对比或区分数据，Highcharts提供了多轴的支持。

Series

数据列。图表上一个或多个数据系列，比如曲线图中的一条曲线，柱状图中的一个柱形。

Tooltip

数据提示框。当鼠标悬停在某点上时，以框的形式提示该点的数据，比如该点的值，数据单位等。数据提示框内提示的信息完全可以通过格式化函数动态指定。

Legend

图例。用不同形状、颜色、文字等 标示不同数据列，通过点击标示可以显示或隐藏该数据列。

Credits

图表版权信息。显示在图表右下方的包含链接的文字，默认是Highcharts官网地址。通过指定credits.enabled=false即可不显示该信息。

Exporting

导出功能按钮。通过引入exporting.js即可增加图表导出为常见文件功能。

PlotLines

标示线（或辅助线）。可以在图表上增加一条标示线，比如平均值线，最高值线等。

PlotBands

标示区域（分辨带）。可以在图表添加不同颜色的区域带，标示出明显的范围区域。

（正文完，最后更新时间：2014-11-21 09:23:29）

图表配置（Chart）

本节主要讲解图表配置，对应的 API 位置为 `chart` ，主要内容包括图表全局样式、绘图区、图表事件、等相关内容。

一、图表容器

Highcharts 实例化中绑定容器的方式有两种 1、通过 dom 调用 highcharts() 函数的方式

```
$("#container").highcharts({
    // Highcharts 配置
});
```

2、通过 chart.renderTo 来指定

```
var charts = new Highcharts.Chart({
    // Highcharts 配置
    chart : {
        renderTo : "container"  // 注意这里一定是 ID 选择器
    }
});
```

二、图表样式

1、宽度、高度

Highcharts 图表的高度和宽度是根据 DIV 容器的宽高来设定的，即

```
<div id="container" style="width:400px;height:400px"></div>
```

如果容器没有设定宽高，默认是 宽 `400px` ， 高 `400px` ，另外设置容器的 `min-width` 属性可以让 highcharts 自适应宽度，实例：

```
<div id="container" style="min-width:400px;height:400px"></div>
```

特别说明：饼图中可以通过设置宽高来让图形填充满整个容器

2、图表样式

图表样式属性包括 border、backgroundColor、margin、spacing、style等

- 边框：包括 borderColor、borderRadius、borderWidth
- 背景：包括 backgroundColor

- 外边距：包括 margin、marginTop、marginRight、marginBottom、marginLeft
- 内边距：包括 spacing、spacingTop、spacingRight、spacingBottom、spacingLeft
- 其他样式：其他属性例如字体等属性，实例代码

```
chart : {
  style : {
    fontFamily:"",
    fontSize:'12px',
    fontWeight:'bold',
    color:'#006cee'
  }
}
```

另外还可以通过 `chart.className` 来绑定 CSS 类并给定 CSS 样式。

3、图表绘图区

图表绘图区的可配置属性有：

- plotBackgroundColor ： 绘图区背景颜色
- plotBackgroundImage ： 绘图区背景图片
- plotBorderColor ： 绘图区边框颜色
- plotBorderWidth ： 绘图区边框宽度
- plotShadow ： 绘图投影

三、事件

1. click ：图表点击事件，效果见 在线演示
2. load ：图表加载完后事件，效果见 在线演示
3. addSeries ：图表增加序列事件，效果见在线演示
4. drilldown ：图表下钻事件，效果见 在线演示
5. drillup ： 图表上钻事件，效果见 在线演示
6. redraw ：图表重绘事件，效果见 在线演示
7. selection ： 图表范围选择事件，效果见在线演示
8. afterPrint ： 图表打印前事件
9. beforePrint ： 图表打印后事件

四、其他配置

1、图表类型

通过 `chart.type` 来指定图表类型，表示如果默认图表类型，即如果 series 中没有指定 type， 那么图表的类型就由 该属性来确定。highcharts 支持的所有图表类型见 `plotOptions` 。

2、图表缩放

图表缩放包括缩放（zoom）和平移（pan），对应的属性有：

- **zoomType** ：缩放类型，可以是水平缩放、竖直缩放、平面缩放，对应的时设置 `zoomType` 为 `"x"` 、 `"y"` 、 `"xy"`
- 缩放恢复按钮：可以指定按钮的样式、位置等，见 resetZoomButton，按钮的文字可以通过 lang 中的属性来指定
- **selectionMarkerFill** ：
- **panning** ：是否启用平移，启用平移后，按住平移键既可以使用鼠标对图表进行平移操作
- **panKey**：平移键，默认是 “Shift”，即在启用平移后，按住指定的按键即可对图表进行平移操作，在线实例

3、3D 属性

Highcharts 4.0 开始支持 3D 图表类型，目前支持 3D 柱形图、3D 饼图、3D 散点图。 3D 相关属性见：chart.options3d ，关于 3D 图形的详细教程将以单独文章形式给出。

4、其他

- 图表反转 ：图表反转指的是将图表的 x轴和 y轴进行对调操作，对应的只需要设置 `chart.inverted = true` 即可。
- 图表动画 ：`chart.animation` 可以设置图表的全局动画效果，这里的动画指的是图表更新时的动画效果，而图表初始化的动画是在 `plotOptions.series.animation` 中启用和关闭的。
- 图表自适应 ：前面说过通过设置图表容器的 `min-width` 可以让图表自适应，这个开关对应的属性是 `chart.reflow` ，另外，还可以通过 API 接口 `Chart.reflow` 在外部对图表进行自适应操作，实例见这里

（正文完，最后更新时间：2015-08-09 18:01:50）

标题（Title）

标题默认显示在图表的顶部，包括标题和副标题（subTitle），其中副标题是非必须的。设置标题和副标题的示例代码如下：

```
title: {
    text: '我是标题'
},
subtitle: {
    text: '我是副标题'
}
```

一、标题的常用属性

标题只有一些文字信息，所以标题的配置无非是一些定位、字体大小、颜色等的配置，常见属性如下表所示：

属性名	描述	默认值
text	标题的文字	"Chart title"
align	文字水平对齐方式，有left、center、right可选	"center"
verticalAlign	文字垂直对齐方式，有top、middle、bottom可选	""
useHTML	是否解析html标签，设置解析后，可以使用例如a等html标签	false
floating	是否浮动，设置浮动后，标题将不占用图表区位置	false
margin	标题和图表区的间隔，当有副标题时，表示标题和副标题之间的间隔	15
style	文字样式，可以设置文字颜色、字体、字号，注意和css有略微的不同，例如font-size用fontSize、font-family用fontFamily表示	{ color: '#3E576F', fontSize: '16px'}
x	相对于水平对齐的偏移量，可以是负数，单位是px	0
y	相对于垂直对齐的偏移量，可以使负数，单位是px	0

更多关于标题的属性请参考API文档：title

二、动态设置和获取标题

1、获取标题内容

可以通过Highcharts对象获取标题内容，实例代码如下

```
var chart = new Highcharts.Chart(options);    // Highcharts构造函数
var title  = chart.title.textStr;             // 通过chart对象获取标题内容
```

2、动态设置标题

Highcharts提供了 `setTitle()` 函数供动态设置标题用， `setTitle()` 函数结构如下

`setTitle (Object title, object subtitle, Boolean redraw)`

参数说明：

- title : 标题对象
- subtitle: 副标题对象
- redraw：是否重绘，即设置标题后是否重新绘制图表，默认是false

实例说明：

```
var title = {
    text:"我是新标题",
    style:{
        color:"#ff0000"
    }
};

var chart = new Highcharts.Chart();
chart.setTitle(title);
```

上述方法不仅仅是设置标题的文字，有的时候我们可能只需要更改标题的样式，例如颜色、字号的，也可以通过该函数实现，示例代码如下

```
var subtitle = {
    style:{
        color:"#000",
        fontWeight:"bold"
    }
};

var chart = new Highcharts.Chart();
chart.setTitle(null,subtitle);     //设置副标题，第一个参数设置为null
```

在线试一试

三、关于标题的常见问题

1、如何在标题中添加链接

实现方法：

设置useHTML为true，然后在标题文字中加入a标签

实例：

```
title :{
    useHTML:true,
    text:"Highcharts中文网 | <a href='http://www.hcharts.cn' target='_blank'>中文教程</a>"
}
```

2、如何隐藏（不显示）标题

实现方法：

设置标题文字为空即可

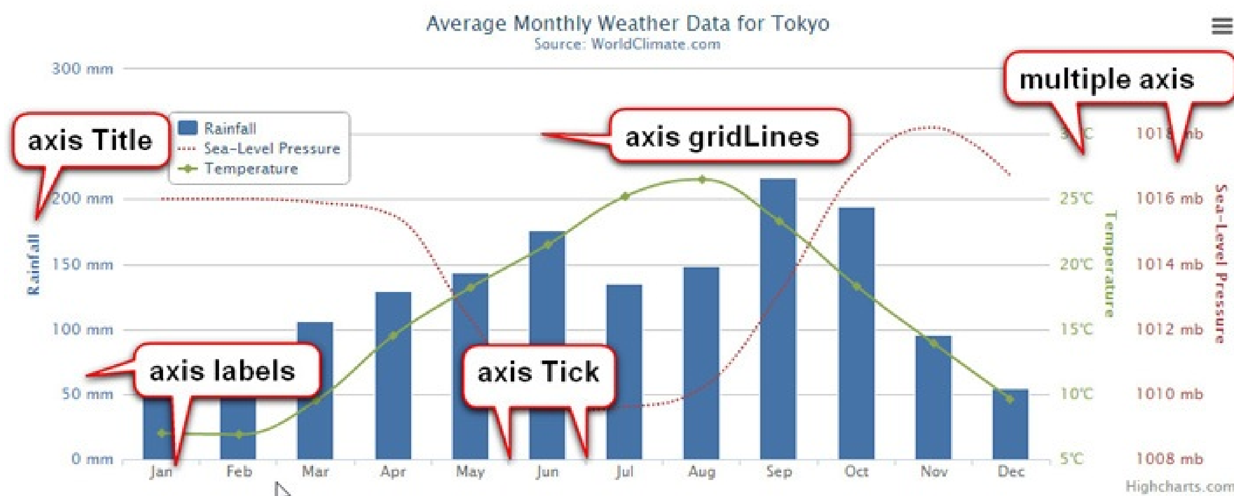
实例：

```
title :{
    text:null
}
```

（正文完，最后更新时间：2015-06-10 11:09:59）

坐标轴（Axis）

所有的图表除了饼图都有X轴和Y轴，默认情况下，x轴显示在图表的底部，y轴显示在左侧（多个y轴时可以是显示在左右两侧），通过设置 `chart.inverted = true` 可以让x，y轴显示位置对调。下图为图表中坐标轴组成部分



图表坐标轴组成部分

一、坐标轴组成部分

1、Axis Title

坐标轴标题。默认情况下，x轴为 `null` （也就是没有title），y轴为 `'Value'` ，设置坐标轴标题的代码如下：

```
xAxis:{
   title:{
       text:'x轴标题'
   }
}
yAxis:{
   title:{
       text:'y轴标题'
   }
}
```

更多关于Axis Title属性请查看API文档相关内容 xAxis.title、yAxis.title。

2、Axis Labels

坐标轴标签（分类）。Labels常用属性有 `enabled` 、 `formatter` 、 `setp` 、 `staggerLines`

1）enabled

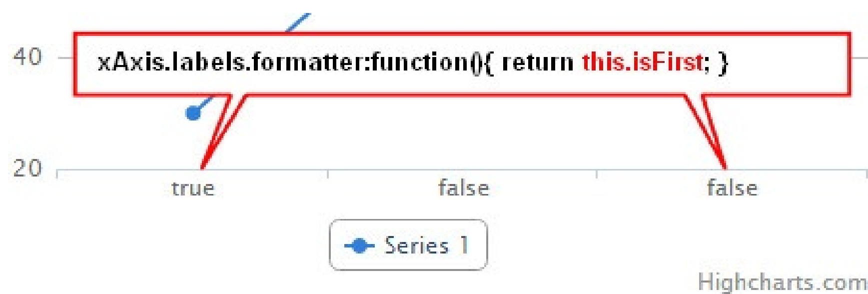
是否启用Labels。x，y轴默认值都是 `true` ，如果想禁用（或不显示）Labels，设置该属性为 `false` 即可。

2）Formatter

标签格式化函数。默认实现是：

```
formatter:function(){
    return this.value;
}
```

`this.value` 代码坐标轴上当前点的值（也就是x轴当前点的x值，y轴上当前点的y值），除了 `value` 变量外，还有 `axis` 、 `chart` 、 `isFirst` 、 `isLast` 可用。例如调用this.isFirst的结果如下图所示



Custom Labels formatter 1

另外一个例子，实现更高级的自定义格式化函数，截图如下：



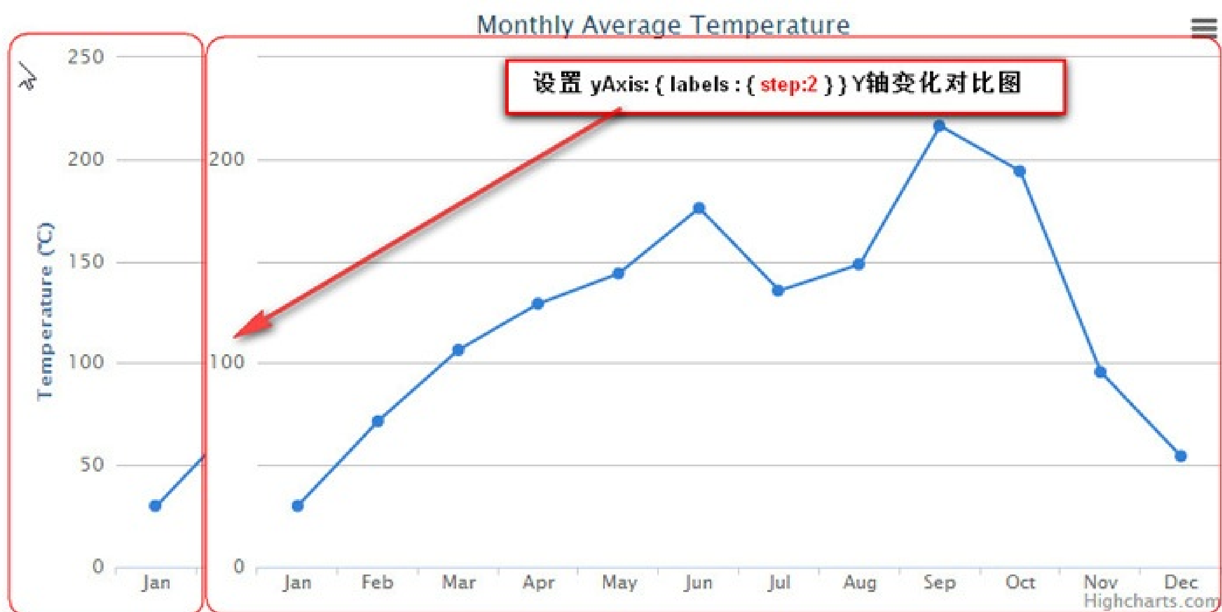
Custom Labels formatter 2

实现代码如下：

```
yAxis: {
  labels: {
    formatter:function(){
      if(this.value <=100) {
        return "第一等级("+this.value+")";
      }else if(this.value >100 && this.value <=200) {
        return "第二等级("+this.value+")";
      }else {
        return "第三等级("+this.value+")";
      }
    }
  }
},
```

3）Step

Labels显示间隔，数据类型为number（或int）。下图说明了step的用法和作用



Y Lables width step

4）staggerLines

水平轴Lables显示行数。（该属性只对水平轴有效）当Lables内容过多时，可以通过该属性控制显示的行数。和该属性相关的还有maxStaggerLines属性。

更多关于Lables的属性请查看API文档 xAxis.labels、yAxis.labels

3、Axis Tick

Tick为坐标轴刻度。默认情况下x轴刻度高(`tickLength` 属性)为5px，宽为1px；y轴宽为0px(也就是不显示刻度)。Tick相关的属性主要有 `tickLength` 、 `tickWidth` 、 `tickColor` 、 `tickInterval` 、 `tickmarkPlacement` 。

1）tickLength、tickWidth、tickColor

分别代表刻度线的长度、宽度、颜色。

2）tickInterval

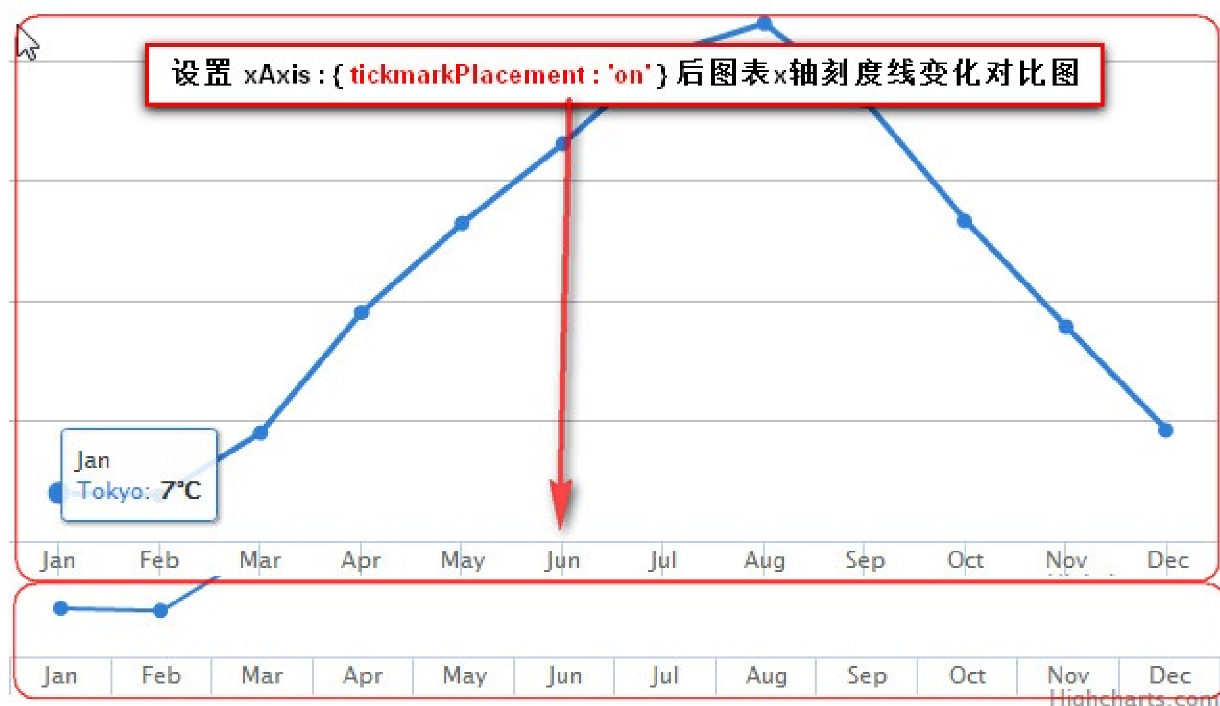
刻度间隔。其作用和 `Lables.step` 类似，就是不显示过多的x轴标签内容，不同的是，`tickInterval` 是真正意义上的调整刻度，而 `Lables.step` 只是调整Labels显示间隔。所以在实际应用中，`tickInterval` 用的多。

针对不同数据类型的坐标轴有不同的默认值。对于线性数据和 `Datetime` 类型数据，其默认值是tickPixelInterval值，对于 `Categorty` 表示间隔一个category。

关于 `tickInterval` 的作用，请访问博客中相关内容《如何处理Highcharts X轴数据过多时显示问题》。

3）tickmarkPlacement

刻度线对齐方式，有 `between` 和 `on` 可选，默认是 `between` 。设置为 `on` 后的变化如下图：



xAxis tickmarkPlacement on

更多关于Tick的属性请查看API文档。

4、Axis gridLines

坐标轴网格线。默认情况下，x轴网格线宽度为0,y轴网格线宽度为1px。网格线共有三个属性可设置，分别是: `gridLineWidth` 、 `gridLineColor` 、 `gridLineDashStyle`

1）gridLineWidth

网格线宽度。x轴默认为0，y轴默认为1px。

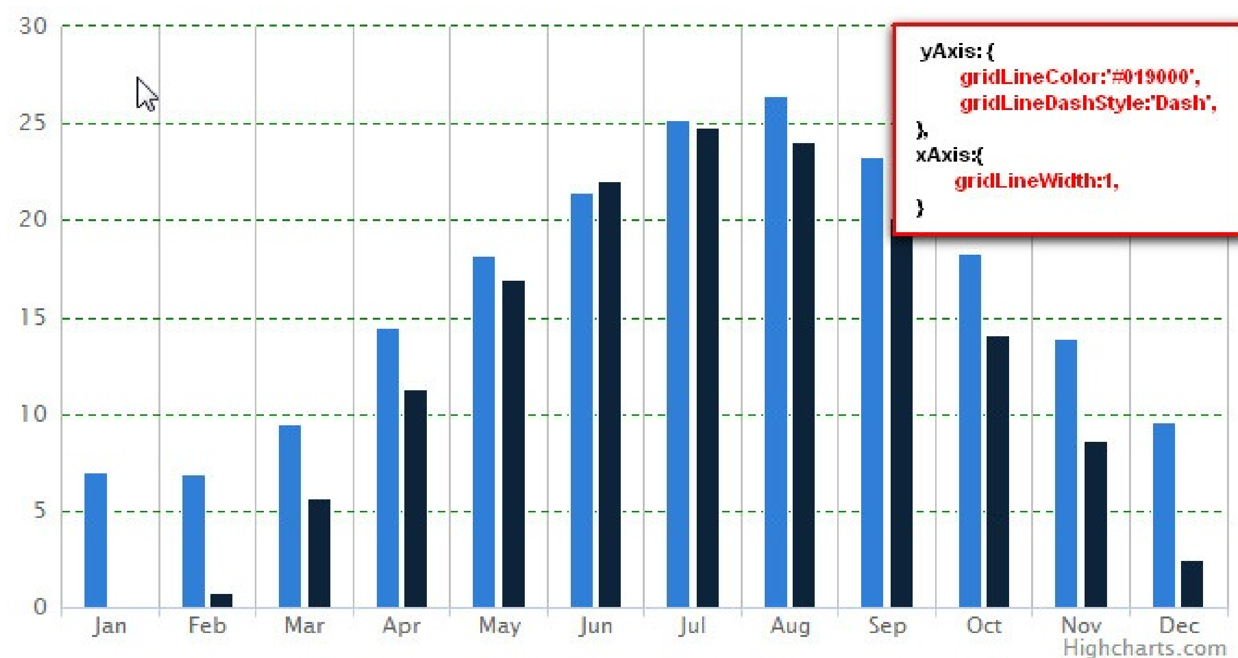
2）gridLineColor

网格线颜色。默认为：`#C0C0C0`。

3）gridLineDashStyle

网格线线条样式。和Css border-style类似，常用的有：`Solid`、`Dot`、`Dash`。

下图为自定义x和y轴的gridLines效果图



Custom gridLines

4、multiple Axis

多个轴。在Highcharts中， 坐标可以是多个，最常见的是多个y轴。多轴存在时，Axis是一个数组，而在赋值时，通过Axis数组的下标与数据关联。如下图所示：

```
yAxis: [{                    // yAxis 1
    ...
    opposite: true
},{                          // yAxis 2
     ...
},{                          // yAxis 3
     ...
     opposite: true
}],

series: [{
    type:'line',
    yAxis: 2,
    data: [49.9, 71.5, 106.4, 129.2, 144.0, 176.0, 135.6, 148.5, 216.4, 194.1, 95.6, 54.4],
}, {
    type:'spline'
    yAxis: 1,
    data: [1016, 1016, 1015.9, 1015.5, 1012.3, 1009.5, 1009.6, 1010.2, 1013.1, 1016.9, 1018.2, 1016.7],
}, {
    yAxis:3,
    type:'column',
    data:[7.0, 6.9, 9.5, 14.5, 18.2, 21.5, 25.2, 26.5, 23.3, 18.3, 13.9, 9.6],
}]
```

1、轴数据通过Axis数组下标关联
2、opposite:true 表示显示位置反转，即y轴显示在右侧，x轴显示在上方
3、在seris数组中，可以指定图表类型(type:'spline')，这样就构成混合图

About multiple Axis

总结如下：

- `Series` 中设置每个轴值时，用轴数组下标关联。
- 设置 `opposite: true` 表示该轴位置反转，即为y轴时显示在右侧，为x轴时显示在顶部（和正常情况下x轴在下，y轴左构成反转）。
- 在 `Series` 中可以设置该轴的类型，多个轴不同的类型（例如：type:'spline'），就构成了多种图表并存的混合图。

更多关于多轴、混合图请查看在线演示平台的 两个(Y)轴的混合图、多个(Y)轴的混合图。

二、坐标轴类型

坐标轴中，可以通过 `Type` 指定坐标轴类型，有 `linear` 、 `logarithmic` 、 `datetime` 、 `category` 可选，默认是： `linear` 。指定类型的实例代码如下：

```
// The types are 'linear', 'logarithmic' and 'datetime'
yAxis: {
    type: 'linear',
}

// Categories are set by using an array
xAxis: {
    categories: ['Apples', 'Bananas', 'Oranges']
}
```

1、linear

线性轴。默认类型，x轴按照 `Axis.tickInterval` 值增长，y轴默认是自适应。

2、logarithmic

对数轴。按照数学中的对数增长，例如1,2,4,8... 用的不多，主要用于对数图表，实例请查看在线演示平台的 对数直线图。

3、datetime

时间轴。时间使用和Javascript 日期对象一样，即用一个距1970年1月1日0时0分0秒的毫秒数表示时间，也就是时间戳。更多Javascript 日期对象请阅读 W3C school 相关内容。

Highcharts有很多时间格式化函数，列举如下：

1）Date.getTime()

获取当前时间戳。实例用法如下：

```
time = Date.getTime();   //time = 1384442746960 (ms)  当前时间为 2013-11-14 23:25:46
```

2）Date.UTC(year,month,day,hours,minutes,seconds,millisec)

通过UTC方式获取指定时间的毫秒数，例如获取 2013-11-14 00:00:00的毫秒数代码如下：

```
time = Date.UTC(2013,11,14,0,0,0,0);  // time = 1386979200000 (ms);
```

3）Highcharts.dateFormat(String format)

Highcharts时间格式化函数，同 PHP格式化函数。具体用法参考API文档 Highcharts.dateFormat()），当然，在本教程的《函数使用》章节中具体讲解。

4、category

数组轴。用的最多也最简单，这里就不多说，更多请查看在线演示平台例子。

三、动态更新及其他相关属性

1、动态更新

坐标轴可以通过函数实现动态更新，在图表绘制完毕后，你可以任意的对你更改而不需要重绘。所有相关函数都在 Axis 中，本教程将在《函数使用》章节具体讲解。

2、其他相关属性

出了Axis中的属性可以对坐标轴有影响外，还有其他属性也可以对其起作用。列举如下：

1）reversed

图表反转，即 `Chart.reversed` ，当其值设置为 `true` 时，x轴和y轴显示的位置对调。详情请查看在线演示平台 轴反转的曲线图。

2）allowDecimals****

控制数轴是否显示小数。

3）min、max****

控制数轴的最小值和最大值。

注意：控制allowDecimals、min、max 属性你可以轻松控制数轴的显示范围等（这也是很常见的问题）

4）opposite

轴倒置。和reversed不同的是，倒置是将轴倒置而不是x、y对调。例如y轴倒置的结果是y轴是从最大的值开始的，最小值反而在最下方。

5）plotLines

标示线，详见：http://www.hcharts.cn/docs/index.php?doc=basic-plotLines

6）plotBands

标示区域，详见：http://www.hcharts.cn/docs/index.php?doc=basic-plotBands

---

（正文完，最后更新时间：2014-06-24 19:18:07）

数据列（Series）

> 数据列配置是 Highcharts 最复杂也是最灵活的配置，如果说 Highcharts 是灵活多变，细节可定制的话，那么数据列配置就是这个重要特性的核心

一、什么是数据列

数据列是一组数据集合，例如一条线，一组柱形等。图表中所有点的数据都来自数据列对象，数据列的基本构造是：

```
series : [{
    name : '',
    data : []
}]
```

提示：数据列配置是个数组，也就是数据配置可以包含多个数据列。

数据列中的 `name` 代表数据列的名字，并且会显示在数据提示框（Tooltip）及图例（Legend）中。

二、数据列中的数据

在数据列的 `data` 属性中，我们可以定义图表的数据数组，通常有三种定义方式：

1、数值数组。在这种情况下，配置数组中的数值代表 Y 值，X 值则根据 X 轴的配置，要么自动计算，要么从 0 起自增，或者是根据 `pointStart` 及 `pointInterval` 自增；在分类轴中，X 值就是 `categoies` 配置，数值数组配置实例如下：

```
data : [1, 4, 6, 9, 10]
```

在线试一试

2、包含两个值的数组集合。在这种情况下，集合中数组的第一个值代表 X， 第二个值代表 Y；如果第一个值是字符串，则代表该点的名字，并且 X 值会如 **1** 中所说的情况决定。数组集合的实例：

```
data : [ [5, 2], [6,3], [8,2] ]
```

在线试一试 （注意例子是 x y 轴对调的）

3、数据点对象集合。在这种情况下，集合中元素都是数据点对象，对象中可以配置数据见 `plotOptions.series` 或 plotOptions.{图表类型} 所列。配置实例：

```
data : [{
    name : "point 1",
    color : "#00ff00",
    y : 0
}, {
    name : "Point 2",
    color : "#ff00ff",
    y : 5
}]
```

在线试一试

另外，通过这种方式还可以增加额外变量，详见例子：增加额外变量

三、数据点及标记

在直角坐标图（即常规的包含X、Y轴的图表）中，数据点相当于图表中的一个（x,y）点。数据点的配置可以在数据列中是数据数组里指定。对于其他类型的图表（非直角坐标图），数据点不仅仅表示 X，Y值，例如在范围图中，数据点包含 x，low， high值；在 OHLC（蜡烛柱状图）中，数据点包含 x，open ， high， low， close；在饼图或仪表图中，数据点只表示一个值。

数据点配置适用所有图表，下面的例子说明了如何指定某个点的颜色：

```
series : [{
    data : [ 29,9, 71.5, 106.4,
    {
        y : 200,
        color : "#BF0B23"
    }, 194.1 , 20 ]
}]
```

在 直线图、曲线图、面积图及面积范围图中可以为数据点指定标记，可以是某种形状， 图片等，实例：

```
series : [{
    data: [29.9, 71.5, 106.4, 129.2, 144.0, 176.0, 135.6,148.5,
    {
      y: 216.4,
      marker: {
          fillColor: '#BF0B23',
          radius: 10
      }
    }, 194.1, 95.6, 54.4]
}]
```

更多关于数据点标记见API文档：http://www.hcharts.cn/api/index.php#plotOptions.series.marker

四、数据列配置

数据列共有三个级别的配置，权重从低到高依次如下：

- 配置在 plotOptions.series 中

  对应的 API 为：http://www.hcharts.cn/api/index.php#plotOptions.series 中，针对所有类型图表有效，一般是通用配置。

- 配置在 plotOptions.{图表类型} 中

  对应的 API 为 ：http://www.hcharts.cn/api/index.php#plotOptions 下的指定图表类型，针对当前类型图表有效，一般是某一种图表的通用配置。

- 配置在 series 中

  对应的 API 为：http://www.hcharts.cn/api/index.php#series， 针对当前数据列有效

以上三中方式自上往下权重依次递增的，也就是配置在 series 中的属性会覆盖 plotOptions 中的配置。 Highcharts API 的这种层级关系体现了 API 设计的继承性和灵活性。

相关内容：论坛帖子

下面列举数据列的一些常用属性

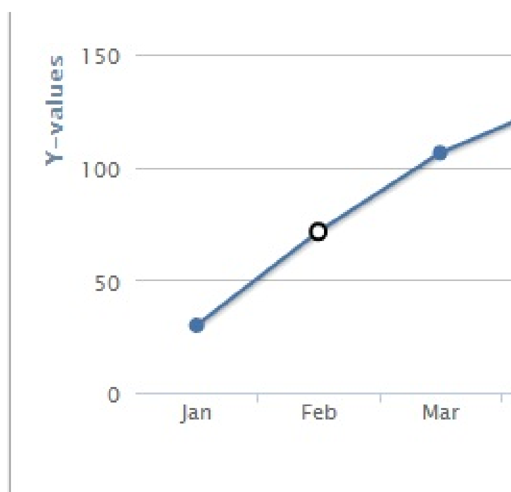
1、动画（Animation）

Highcharts 图表默认是以动画的形式展现图形加载过程的，可以通过 `series.animation` 或 `plotOptions.series.animation` 来指定动画相关配置（是否启用动画，动画效果等）。

2、颜色（Color）

可以通过 series.color 来指定数据列的颜色，通过 plotOptions.{图表类型}.color 来给某一种类型的图表设定颜色。

3、点的选择（point Selection）



通过设置 allowPointSelect = true 可以使数据点可选择

```
plotOptions: {
  series: {
    allowPointSelect: true
  }
}
```

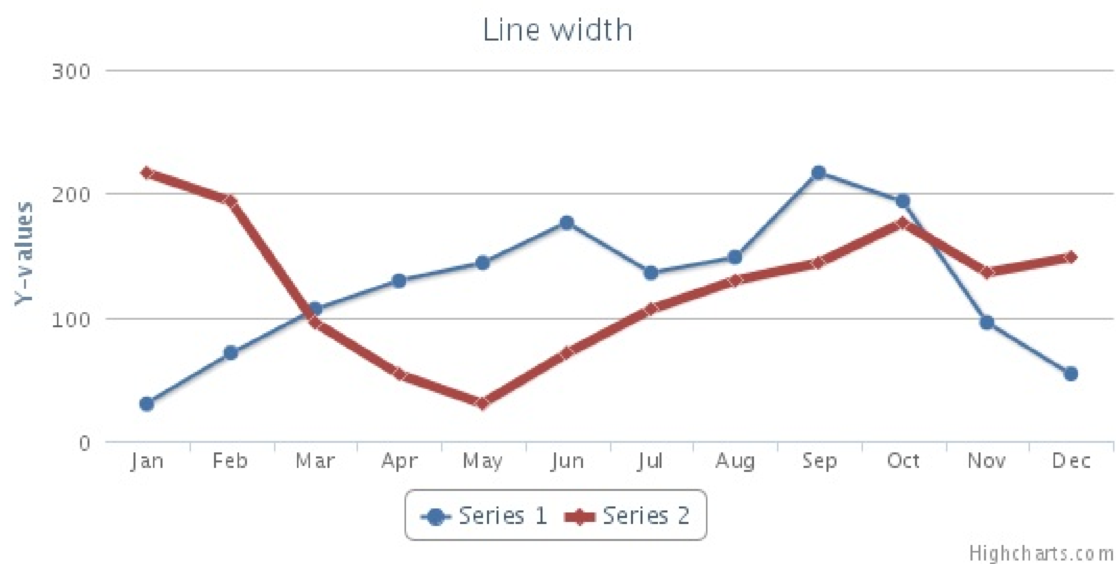
对应的获取选中的点是通过 `chart.getSelectedPoints()` 函数来实现的

```
var selectedPoints = chart.getSelectedPoints();
```

在线试一试

提示：按住 CTRL 或 SHIFT 键可以多选

4、线条宽度（lineWidth）



可以通过 lineWidth 来指定线条宽度

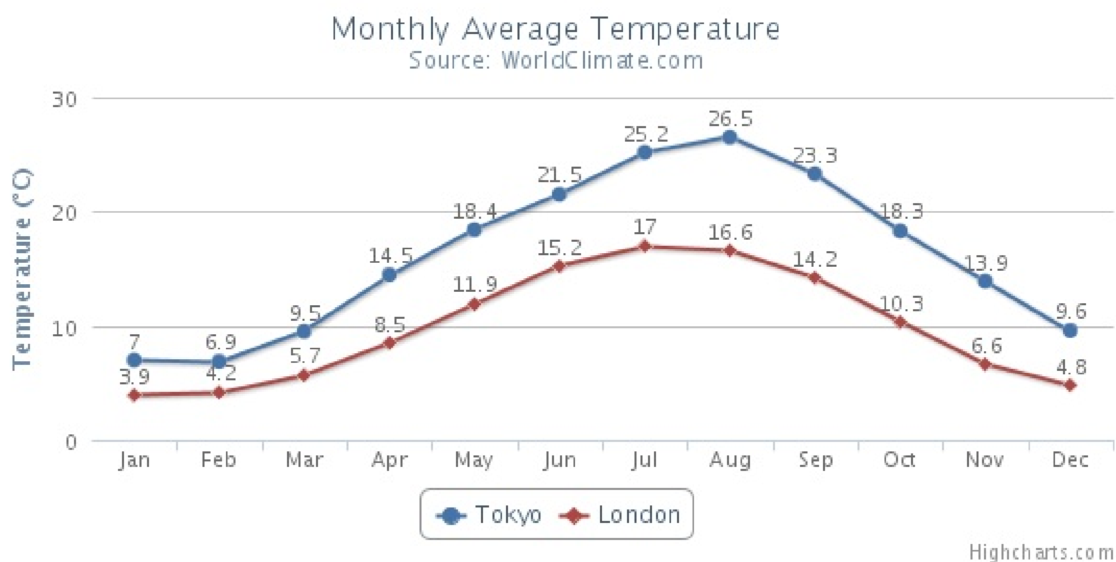
```
series: [{
  data: [216.4, 194.1, 95.6],
  lineWidth: 5
}]
```

在线试一试

5、鼠标形状（cursor）

cursor 属性可以指定鼠标形状，即指定当鼠标悬停在数据列上时对应的鼠标样式（当配置了数据列点击事件时）。

6、数据标签（dataLables）



数据标签指的是在数据点上显示一些数据信息标签，对应的 API 为 http://www.hcharts.cn/api/index.php#series.data.dataLabels

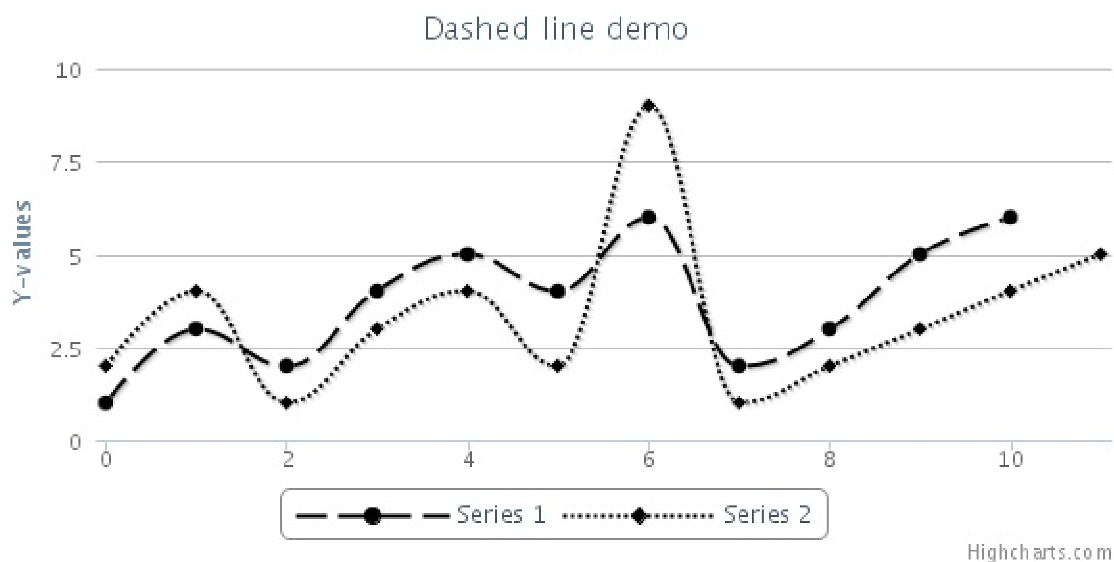
```
plotOptions: {
    line: {
        dataLabels: {
            enabled: true
        }
    }
}
```

数据标签默认显示当前数据点的点值，可以通过 `formatter` 函数或 `format` 来对其格式化。

```
plotOptions: {
    line: {
        dataLabels: {
            enabled: true,
            formatter: function() {
                return this.x + "     " + this.y;
            },
            // format: "{x}      {y}"
        }
    }
}
```

在线试一试

7、线条样式（Dash Style）



dashStyle 可以指定线条的样式

```
series: [{
    data: [1, 3, 2, 4, 5, 4, 6, 2, 3, 5, 6],
    dashStyle: 'longdash'
}]
```

在线试一试

8、zones

我们经常会遇到这样的需求：用不同颜色标识出不同范围的值，例如 90-100 用绿色表示，60-80 用蓝色表示，小于 60用红色表示。在 Highcharts 4.1 之前，我们可以通过 plotBands 来标识出不同范围对应的背景（效果见实例），或者用 plotLine 画一条标识线（见教程），还可以用不同颜色标记出点的颜色，这些解决方案都有自己的用途，但在某些情景下并不是最优方案。

Highcharts 4.1 增加了一个非常牛逼的新特性：Zones，先来看个例子：

Result Javascript Html Edit in HCode

对应的代码也很简单：

```
$(function() {
    $('#container').highcharts({
        series: [{
            data: [-10, -5, 0, 5, 10, 15, 10, 10, 5, 0, -5],
            zones: [{
                value: 0,
                color: '#f7a35c',
                dashStyle: 'dot'
            }, {
                value: 10,
                color: '#7cb5ec'
            }, {
                color: '#90ed7d'
            }]
        }]
    });
});
```

在线试一试

可以看到 Zones 是个数组，常见的属性有 value，color，dashStyle

- value 表示对超过这个值的区域有效
- color 对当前范围设置颜色
- dashStyle 对当前范围设置线条颜色
- fillColor 对当前范围设置填充颜色（针对面积图）

zones 默认的是针对 Y 轴，可以通过 zoneAxis = x 来指定当前配置是针对 x 轴，实例：

```
$(function() {
    $('#container').highcharts({
        title: {
            text: 'Zone with dash style'
        },
        subtitle: {
            text: 'Dotted line typically signifies prognosis'
        },
        xAxis: {
            categories: ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', '
        },
        series: [{
            data: [29.9, 71.5, 106.4, 129.2, 144.0, 176.0, 135.6, 148.5, 216.4, 194.1, 95
            zoneAxis: 'x',
            zones: [{
                value: 8
            }, {
                dashStyle: 'dot'
            }]
        }]
    });
});
```

在线试一试

提示：

1、对于 x 轴的 zones 的 value 值表示的是 x 轴下标 2、zones 数组里的每个对象具有继承性，相同的属性会覆盖前一个配置

在 Highcharts 中，数据列的配置是个非常重要的配置，同时又由于可配置的属性非常对，配置的灵活性非常高，需要大家自己自己体会和理解，关于数据陪的详细配置将会在【图表类型章节详细说明】

（正文完，最后更新时间：2015-09-10 13:45:09）

颜色（colors）

本节详细说明了图标中线条颜色、文字颜色等和颜色相关的内容。

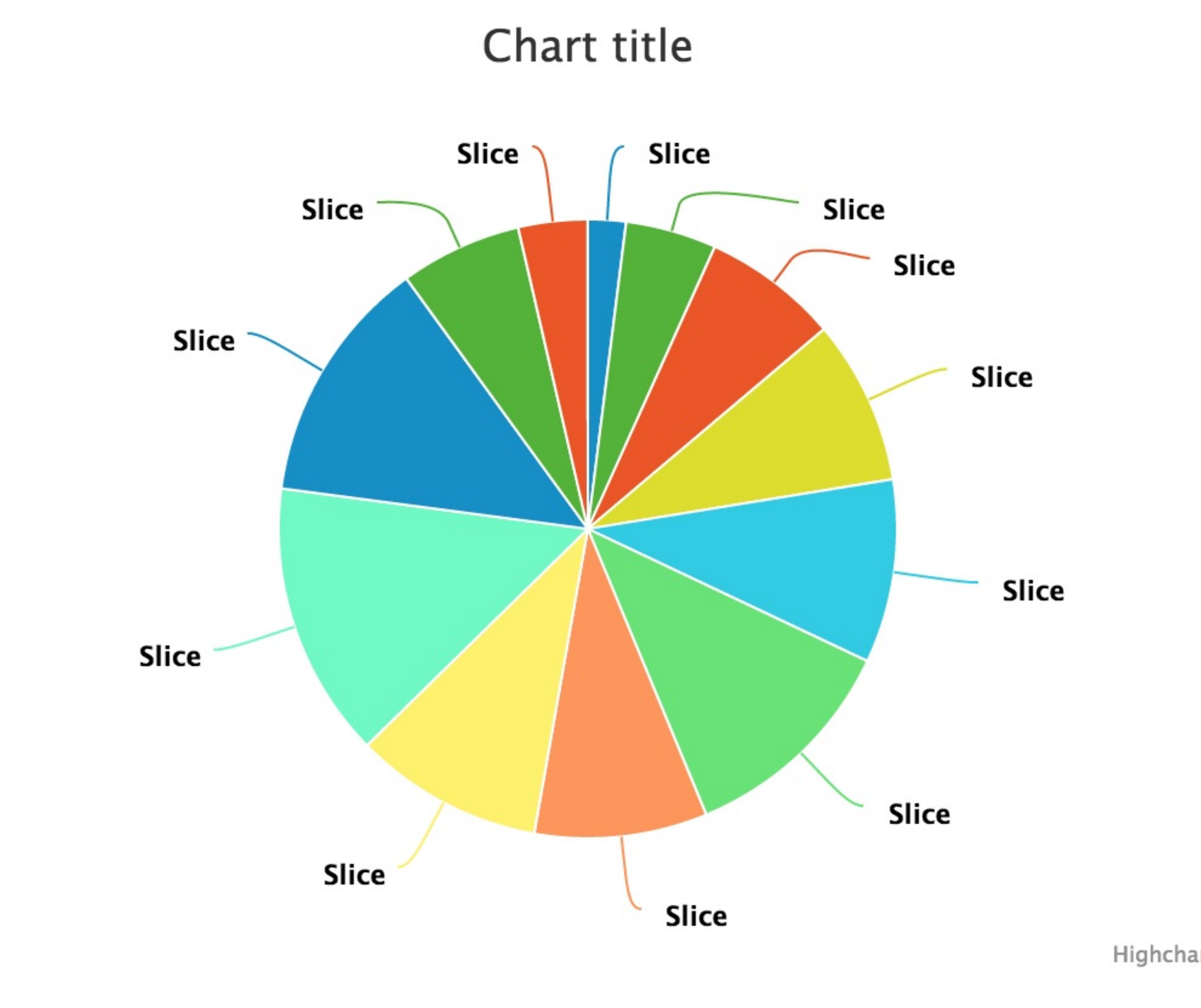
一、数据列颜色

Highcharts 中数据列的颜色是通过 `colors` 来指定的，colors 是个颜色值数组，默认是：

```
colors: ['#7cb5ec', '#434348', '#90ed7d', '#f7a35c', '#8085e9',
        '#f15c80', '#e4d354', '#8085e8', '#8d4653', '#91e8e1']
```

共有 10个默认颜色，你可以修改颜色值或增加颜色个数来自定义图表数据列颜色。

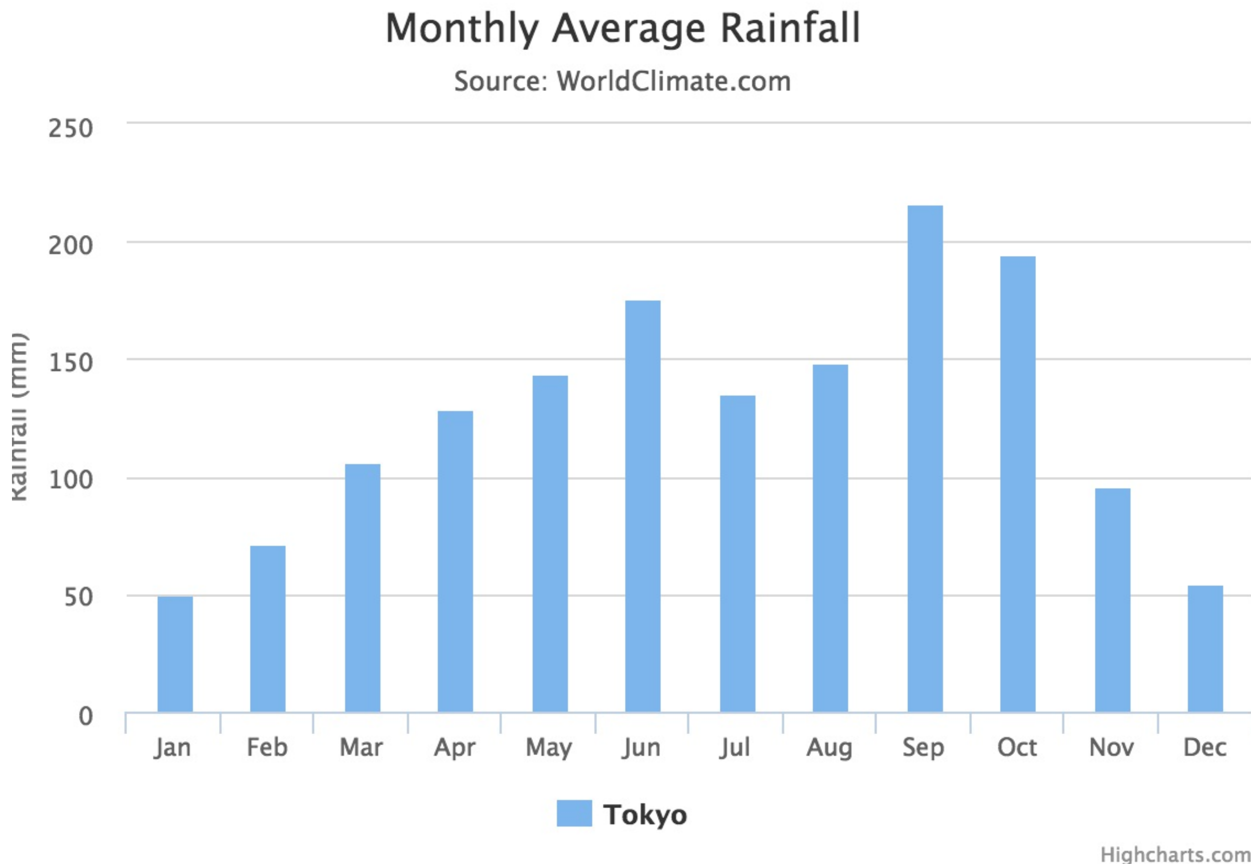
数据列调用颜色的方式是第 n 个数据列使用第 n 个颜色数组里的值，当序列的数量超过颜色数组的长度，后续的序列将会从头调用。



在线试一试

二、柱形图的颜色

柱形图（包括柱状图、条形图等）里一组柱形颜色是一样的，很多人对此表示不理解




```
series: [{
  name: 'Tokyo',
  data: [49.9, 71.5, 106.4, 129.2, 144.0, 176.0, 135.6, 148.5, 216.4, 194.1, 95.6, 54.4]
}]
```

通过代码我们知道，一组柱形是属于同一个数据列的，所以他们的颜色当然时一样的。

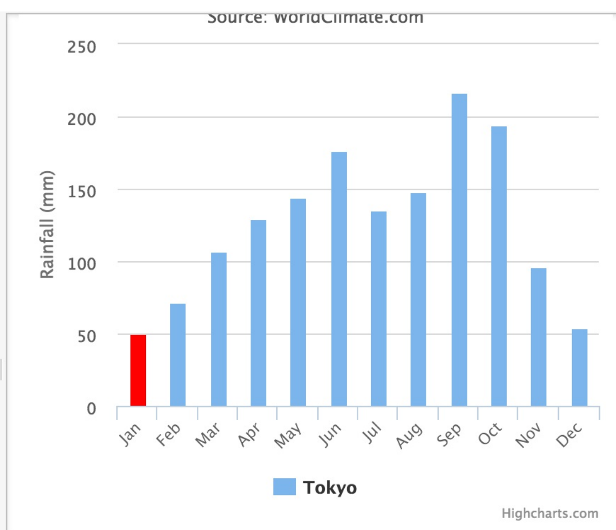
你可能想到下面的这种方法：

```
series: [{
    name: "Column series",
    data: [{
        y:49.9,
        color:"#ff0000"
    },
    // ... 省略代码
    ]
}]
```

这是在上一节“数据列”中说到的颜色赋值方式，用这种方法虽然可以实现想要的效果，但是如果分别对每个数据列赋值，显然不合理。

```javaScript
            shared: true,
            useHTML: true
        },
        plotOptions: {
            column: {
                pointPadding: 0.2,
                borderWidth: 0
            }
        },
        series: [{
            name: 'Tokyo',
            data: [{
                y:49.9,
                color:'#ff0000'
            } , 71.5, 106.4, 129.2, 144.0,
176.0, 135.6, 148.5, 216.4, 194.1, 95.6, 54.4]

        }]
    });
});
```



highcharts 直接提供对柱形图同数据列设置颜色，属性是：colorByPoint

API 给出的说明是：

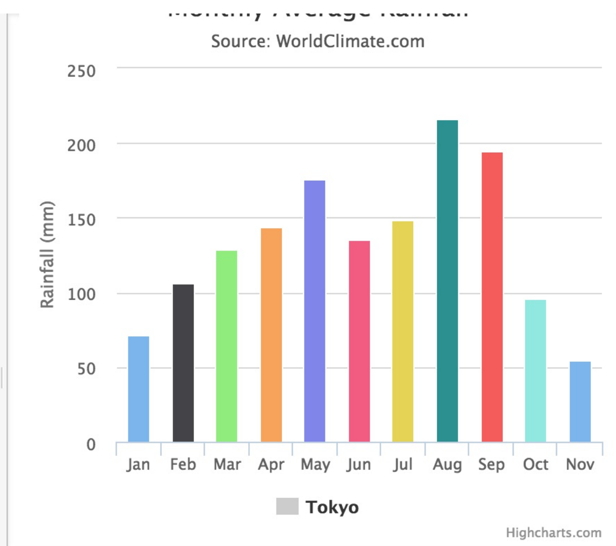
```
this option determines whether the chart should receive one color per series or one color
```

即 colorByPoint 决定了图表是否给每个数据列或每个点分配一个颜色，默认值是 false， 即默认是给每个数据类分配颜色， 设置为 true 则是给每个点分配颜色。

设置 colorByPoint = true 后的效果见下图：

```javaScript
            footerFormat: '</table>',
            shared: true,
            useHTML: true
        },
        plotOptions: {
            column: {
                pointPadding: 0.2,
                borderWidth: 0
            }
        },
        plotOptions:{
            column:{
                colorByPoint:true
            }
        },
        series: [{
            name: 'Tokyo',
            //colorByPoint:true,   或者直接写在数据列里
            data: [71.5, 106.4, 129.2, 144.0,
176.0, 135.6, 148.5, 216.4, 194.1, 95.6, 54.4]

        }]
    });
});
```



在线试一试

其他图形可以用 colorByPoint

通过搜索 API 文档可以知道其他图表也可以用到 colorByPoint 这个属性

- bar 条形图
- column 柱状图

- columnrange 柱状范围图
- heatmap 热力图
- treemap
- waterfall 瀑布图
- boxplot 箱线图

注：API 文档搜索功能很强大，请好好利用

三、图表中文字颜色

图表中所有文字都设置字体、颜色等样式，一般是通过 `style.color` 来设置文字颜色的，style 里可以用的属性如下所示：

```
style: {
    color: '#ff0000',
    fontSize: "12px",
    fontWeight: "blod",
    fontFamily: "Courier new"
}
```

下面列举图表中常见文字的配置位置：

1. 标题样式：title.style 及 subtitle.style
2. 坐标轴标签样式：xAxis.lables.style 及 yAxis.lables.style
3. 图例文字样式：legend.itemStyle
4. 数据提示框文字样式：tooltip.style
5. ....

总之请自行在API文档上搜索，图表中任何文字都是可以精准控制的。

（正文完，最后更新时间：2015-06-03 22:24:06）

数据提示框（Tooltip）

数据提示框指的当鼠标悬停在某点上时，以框的形式提示该点的数据，比如该点的值，数据单位等。数据提示框内提示的信息完全可以通过格式化函数动态指定；通过设置 `tooltip.enabled = false` 即可不启用提示框。



一、提示框外观

下面的实例代码给出了关于数据提示框的外观的常用配置

```
tooltip: {
    backgroundColor: '#FCFFC5',   // 背景颜色
    borderColor: 'black',         // 边框颜色
    borderRadius: 10,             // 边框圆角
    borderWidth: 3,               // 边框宽度
    shadow: ture,                 // 是否显示阴影
    animation: true               // 是否启用动画效果
    style: {                      // 文字内容相关样式
        color: "#ff0000",
        fontSize: "12px",
        fontWeight: "blod",
        fontFamily: "Courir new"
    }
}
```

提示：背景颜色也可以设置为渐变色

在线试一试

二、提示框内容

数据提示框的目的是为了展示数据点相关的数据，具体展示的内容完全可以通过多种灵活的方式来实现。

1、格式化函数

formatter

数据提示框格式化函数，功能最强大也是最灵活的方法，函数里 `this` 关键字代表着当前数据点对象，常用的变量有：

- this.x ： 当前点 X 值
- this.y / this.point[i].y ： 当前点的 Y 指 / 当前第 i 个点的 Y 值 （tooltip 共享的情况下，关于共享见下方说明）
- this.point / this.point[i] ： 当前点 / 当前第 i 个点（tooltip 共享的情况下）
- this.series / this.point[i].series ： 当前数据列 / 当前第 i 个点的数据列（tooltip共享）
- this.total
- this.percentage

所有的可用属性可以通过 `console.log(this)` 来查看

小技巧：通过 console.log() 可以很清楚的看到对象中的所有属性及值，这在调试的时候非常好用。

pointFormatter

数据提示框中单个点的格式化函数。默认是：

```
pointFormatter: function() {
    return '<span style="color:{'+this.series.color+'}">u25CF</span> {'+
            this.series.name+'}: <b>{'+this.y+'}</b><br/>.'
}
```

2、格式化字符串

格式化字符串是格式化函数的简化版，是一种利用特殊符号加变量字符的形式来代替函数的表达式。

headerFormat

数据提示框头部格式化字符，默认是：

```
<span style="font-size: 10px">{point.key}</span><br/>
```

pointFormat

单个点的格式化字符串，实现的内容同 pointFormatter，默认实现是：

```
<span style="color:{series.color}">u25CF</span> {series.name}: <b>{point.y}</b><br/>.
```

对比下 pointFormatter 和 pointFormat 我们可以知道两两种方式的差别：

- pointFormat 更简单，适用于简单的内容格式化
- pointFormatter 更灵活，适用于相对复杂的自定义内容

上述两个观点也就是格式化函数和格式化字符串的优缺点，在使用的时候，请灵活运用。

3、时间格式化

在时间图表中，很常见的一个需求是时间的格式化显示，在数据提示框中，我们可以通过时间格式化来统一时间的显示。

dateTimeLabelFormats

数据框中的时间点的格式化，默认实现是：

```
{
    millisecond:"%A, %b %e, %H:%M:%S.%L",
    second:"%A, %b %e, %H:%M:%S",
    minute:"%A, %b %e, %H:%M",
    hour:"%A, %b %e, %H:%M",
    day:"%A, %b %e, %Y",
    week:"Week from %A, %b %e, %Y",
    month:"%B %Y",
    year:"%Y"
}
```

关于时间格式化中字符的

xDateFormat

数据提示框头部时间格式化字符串。

在线试一试

4、html 内容

数据提示框默认（在没开启支持 HTML 模式的情况下）支持少量的 HTML 标签，包括 `<b>` 、 `<br>` 、 `<strong>` 、 `<i>` 、 `<em>` 、 `<br/>` 、 `<span>` ，标签的内容可以通过 style 属性来指定，不过仅限文字相关的 CSS 样式属性。

通过设置 `tooltip.useHTML = true` 可以开启 HTML 模式，即可以用纯 HTML 内容来渲染数据提示框（默认是以 SVG 渲染）。

开启 HTML 模式后，就可以给提示框添加 链接、图片、表格等 HTML 元素，给提示框添加表格的示例代码是：

```
tooltip: {
    shared: true,
    useHTML: true,
    headerFormat: '<small>{point.key}</small><table>',
    pointFormat: '<tr><td style="color: {series.color}">{series.name}: </td>' +
        '<td style="text-align: right"><b>{point.y} EUR</b></td></tr>',
    footerFormat: '</table>',
    valueDecimals: 2
}
```

在线试一试

5、值的前缀、后缀及小数点

在展现数据信息是，我们经常会给数据添加一些修饰信息，例如数据单位。highcharts 提供了 valuePrefix、valueSuffix 来给数据添加前缀及后缀。

```
tooltip: {
    valuePrefix: '￥',
    valueSuffix: '元'
}
```

另外，对于小数点的处理，可以通过 valueDecimals 来指定保留小数位数（当然可以通过格式化函数来进行更复杂的处理）。

在线试一试

对于多个数据列数据提示框添加后缀时，一般是将属性分别配置在数据列中，实例：

```
series: [{
    name: 'Rainfall',
    type: 'column',
    yAxis: 1,
    data: [49.9, 71.5, 106.4, 129.2, 144.0, 176.0, 135.6, 148.5, 216.4, 194.1, 95.6, 54.4
    tooltip: {
        valueSuffix: ' mm'
    }

}, {
    name: 'Temperature',
    type: 'spline',
    data: [7.0, 6.9, 9.5, 14.5, 18.2, 21.5, 25.2, 26.5, 23.3, 18.3, 13.9, 9.6],
    tooltip: {
        valueSuffix: '°C'
    }
}]
```

在线试一试

6、多个数据列共用一个提示框（**Shared**）

多个数据列的图表中，常常需要对多个数据列的数据做对比，将多个数据列的相同分类同时展示在数据提示框中也是非常常见的需求，Highchart 中， `tooltip.shared` 的作用就是将多个数据分享到同一个数据提示框中，也就是多个数据共用的数据提示框。

在线试一试

三、其他特性

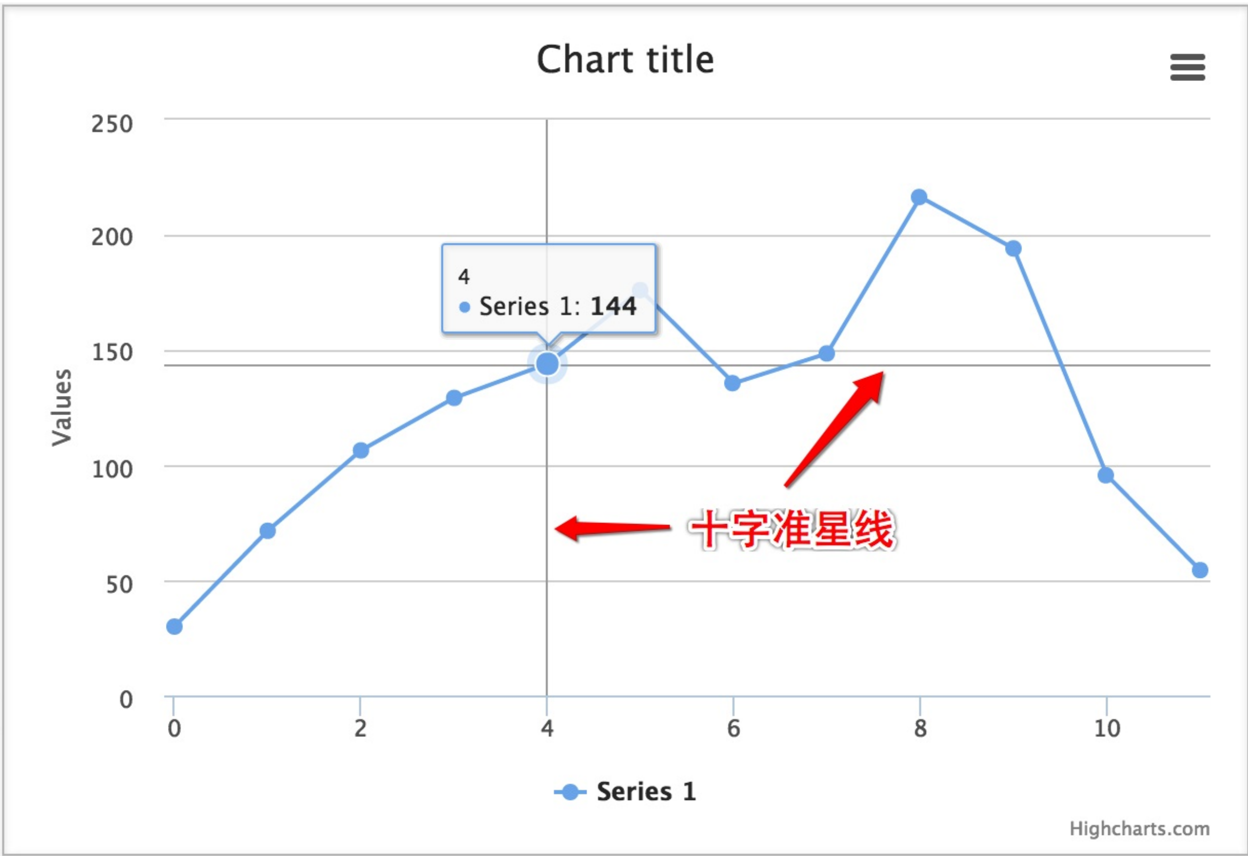
1、十字准星

crosshairs 有三种配置形式，最基础的是设置 crosshairs = true 表示启用竖直方向准星线，三种设置方式是：

```
crosshairs: true          // 启用竖直方向准星线
```

```
crosshairs: [true, true]  // 同时启用竖直及水平准星线
```

```
crosshairs: [{            // 设置准星线样式
    width: 3,
    color: 'green'
}, {
    width: 1,
    color: "#006cee",
    dashStyle: 'longdashdot',
    zIndex: 100
}]
```



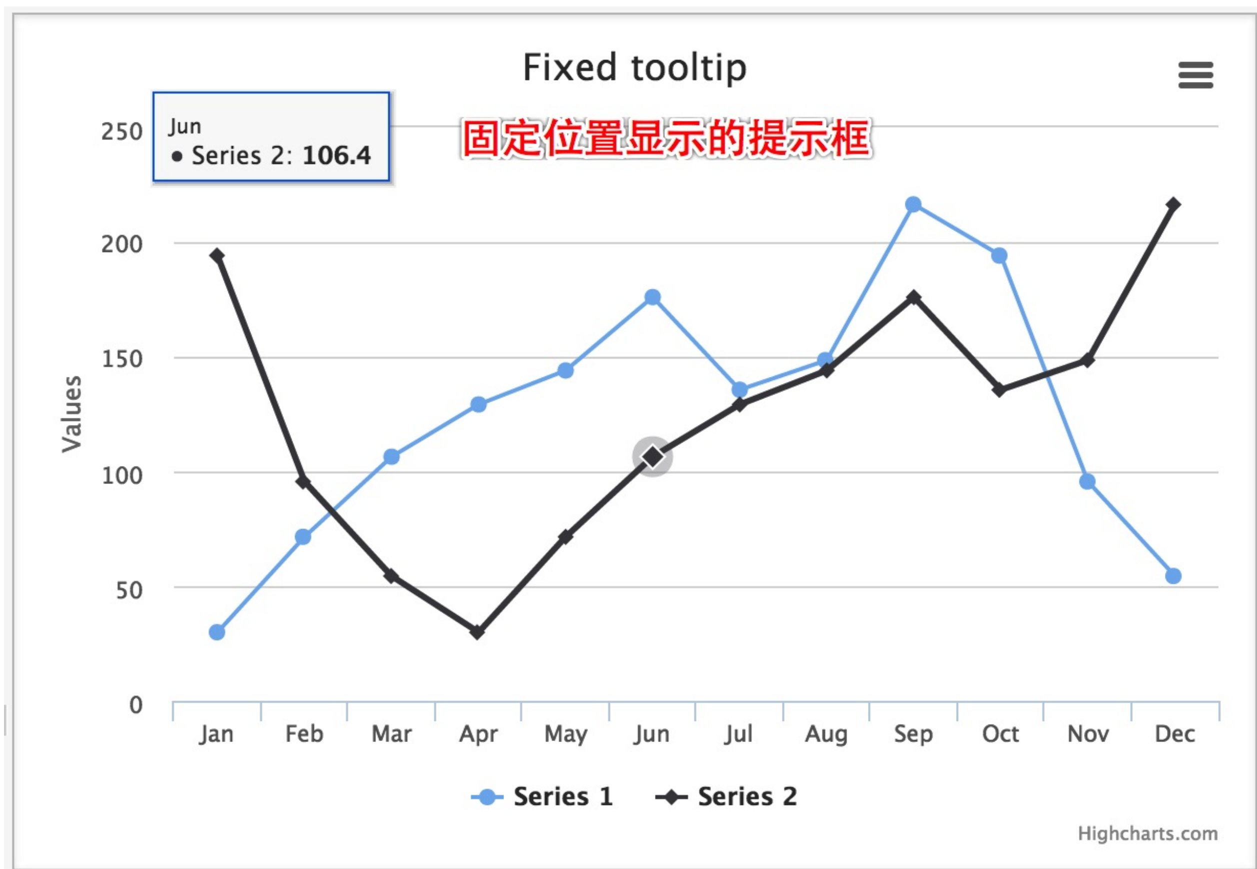
在线试一试

2、固定位置显示提示框

通过配置 `tooltip.positioner` 可以让数据提示框以固定位置显示，实例如下

```
positioner: function() {
    return {
        x: 60,
        y: 80
    }
}
```

其中 x 表示相对图表右上角水平偏移，y 为竖直方向的偏移。



在线试一试

3、鼠标行为

- stickyTracking：设置提示框触发模式，默认是鼠标在点的附近就触发提示框，设置 false 后只有鼠标划过点才触发
- hideDelay：提示框隐藏延迟时间

四、常见问题

1、提示框乱码了怎么办？

通过上面的学习我们知道，数据点的默认格式化内容是：

```
<span style="color:{series.color}">u25CF</span> {series.name}: <b>{point.y}</b><br/>.
```

其中的 “\u25CF” 是一个 Unicode 码，也是导致乱码的根源，最简单的解决办法有两种：

1）将你的页面编码设置 UTF-8

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">    // 设置网页编码
</head>
// ... 省略代码
```

2）重写提示框数据点格式化函数或格式化字符串，例如将 pointFormat 重写为：

```
<span style="color:{series.color}"></span> {series.name}: <b>{point.y}</b><br/>.
```

2、提示框被文字标签覆盖了怎么办？

3、如何在外部触发提示框？

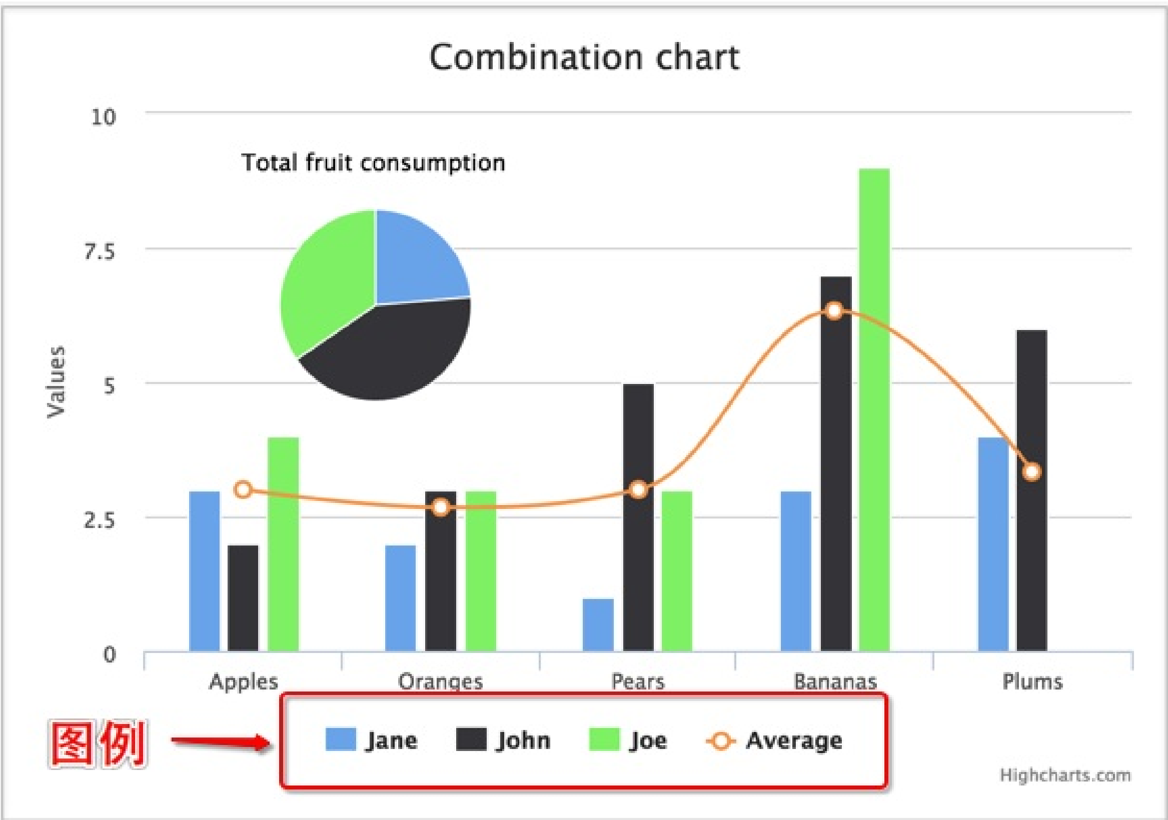
见：论坛帖子

即调用 `chart.tooltip.refresh()` 函数实现（内部没有公开函数，通过研究源码才知道这个函数的，研究源码非常有意义！）

（正文完，最后更新时间：2015-06-09 20:46:02）

图例（Legend）

图例是图表中用不同形状、颜色、文字等 标示不同数据列，通过点击标示可以显示或隐藏该数据列；通过设置 `legend.enabled = true | false` 来打开或关闭图例。



一、图例样式

1、图例容器样式

图例容器指的是整个图例容器的样式，包含背景、边框、边距、宽度等，详细属性及说明见下表

参数名	解释	默认值
backgroundColor	背景颜色	null
borderColor	边框颜色	'#909090'
margin	外边距	15
padding	内边距	8
maxHeight	最大高度	null
navigation	导航，当设置了最大高度后，图例无法完整显示时，则会用导航的形式展示（分页），详见API文档	
shadow	图例阴影效果，赋值可以是 boolean 或 Object，详见API文档	false
width	图例宽度	null
verticalAlign	垂直对齐方式，有 'top'， 'middle' 及 'bottom' 可选	'bottom'
useHTML	是否以HTML形式渲染（默认是SVG渲染），当使用 HTML 模式渲染是，图例导航无效	false

2、图例项样式

上面说到了图例容器的样式可以控制图例整体样式，对应配置图例里的内容是通过图例项相关属性来控制的，见下表

参数名	解释	默认值
itemDistance	图例项间距	20
itemStyle	图例样式	`temStyle: { cursor: 'pointer', color: '#3E576F' }`
itemHiddenStyle	图例隐藏时的样式	`itemHiddenStyle: { color: '#CCC' }`
itemHoverStyle	图例鼠标划过时样式	`itemHoverStyle: { color: '#000' }`
itemMarginBottom	图例项底边距	0
itemMarginTop	图例项顶部边距	0
itemWidth	图例项宽度	null
symbolHeight	图例项标示高度	12
symbolPadding	图例项标示内边距	5
symbolRadius	图例项标示圆角	2
symbolWidth	图例项标示宽度	16

二、图例内容及定位

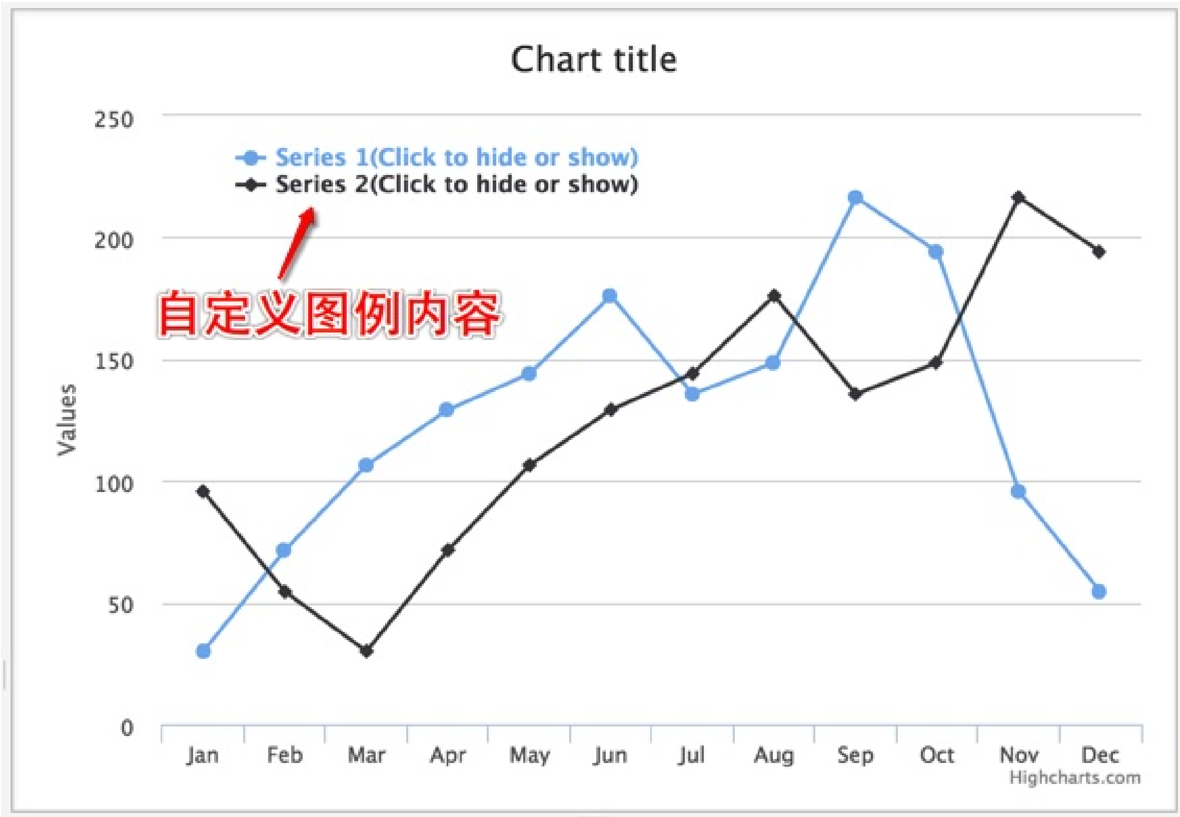
1、图例内容

参数名	解释	默认值
labelFormat	图例文字格式化字符串	`{name}`
labelFormatter	图例内容格式化函数	`function() { return this.name}`
reversed	是否倒序	false
rtl	文字是否显示在符号前面，主要针对阅读习惯从右至左的用户	false
title	图例标题	`title: { text: null, style: { fontWeight: "blod"}}`

关于格式化函数及格式化字符串这里简单说明如下：

```
labelFormatter: function() {
    /*
     *  格式化函数可用的变量：this， 可以用 console.log(this) 来查看包含的详细信息
     *  this 代表当前数据列对象，所以默认的实现是 return this.name
     */
     return  this.name + '(Click to hide or show)';

     `} `
```



在线试一试

labelFormat 格式化字符是格式化函数的一种简写方式，即用包含变量的字符串代替函数。

对于上面格式话函数的代码，完全可以用更简洁的方式实现：

```
labelFormat: '{name} (Click to hide or show)';
```

在线试一试

2、定位

下面是图例位置的确定的一些配置。

参数名	解释	默认值
align	图例在图表中的对齐方式，有 “left”, "center", "right" 可选	“center”
floating	图例是否浮动，设置浮动后，图例将不占位置	false
layout	图例内容布局方式，有水平布局及垂直布局可选，对应的配置值是： “horizontal”， “vertical”	"horizontal"
x	水平偏移	0
y	竖直偏移	0

三、图例事件

图例默认的点击行为是显示或隐藏当前数据列。

```
plotOptions: {
  series: {
    events: {
        legendItemClick: function(e) {
            /*
             * 默认实现是显示或隐藏当前数据列，e 代表事件， this 为当前数据列
             */
        }
    }
  }
}
```

禁用图例点击隐藏效果

```
plotOptions: {
  series: {
    events: {
        legendItemClick: function(e) {
            return false; // 直接 return false 即可禁用图例点击事件
        }
    }
  }
}
```

在线试一试

上述代码对饼图是无效的，API给出的说明如下

> Not applicable to pies, as the legend item is per point. See point.events.

也就是对于饼图对应 legendItemClick 事件是 point.legengItemClick 。

```
plotOptions: {
  pie: {
    point: {
        events: {
            legendItemClick: function(e) {
                return false; // 直接 return false 即可禁用图例点击事件
            }
        }
    }
  }
}
```

在线试一试

改变图例点击默认响应（默认是点击某个图例显示或隐藏当前数据列，这里改变为点击某个图例只显示当前数据列，隐藏其他数据列）

```
plotOptions: {
    series: {
        events: {
            legendItemClick: function(e) {
                var index = this.index;

                var series = this.chart.series;

                if (!series[index].visible) {

                    for (var i = 0; i < series.length; i++) {

                        var s = series[i];

                        i === index ? s.show() : s.hide();
                    }
                }

                return false;
            }
        }
    }
}
```

在线试一试

四、关于图例的常见问题

以下列举出关于图例的常见问题，

1、如何不显示某个数据列的图例

1）设置 showInLegend

```
series: [{
    data: [],
    name: "",
    showInLegend: false // 设置为 false 即为不显示在图例中
}, {
    data: [],
    name: "",
    showInLegend: true // 默认值
}]
```

2）自定义图例格式化函数

```
legend: {
    layout: 'vertical',
    backgroundColor: '#FFFFFF',
    floating: true,
    align: 'left',
    verticalAlign: 'top',
    x: 90,
    y: 45,
    labelFormatter: function() {
        /*
         * 获取数据列下标，通过下标判断做一系列处理
         * 还可以通过获取数据列名字等来做判断（通过 console.log(this) 来查看数据列详细信息）
         */
        var index = this._i;

        // return null 则可以不显示图例项
        return index === 0 ? null : this.name;
    }
}
```

在线试一试

2、图例是否可拖动

可以，利用官方插件“Draggable Legend” 即可实现

3、是否可以在图例中显示数值

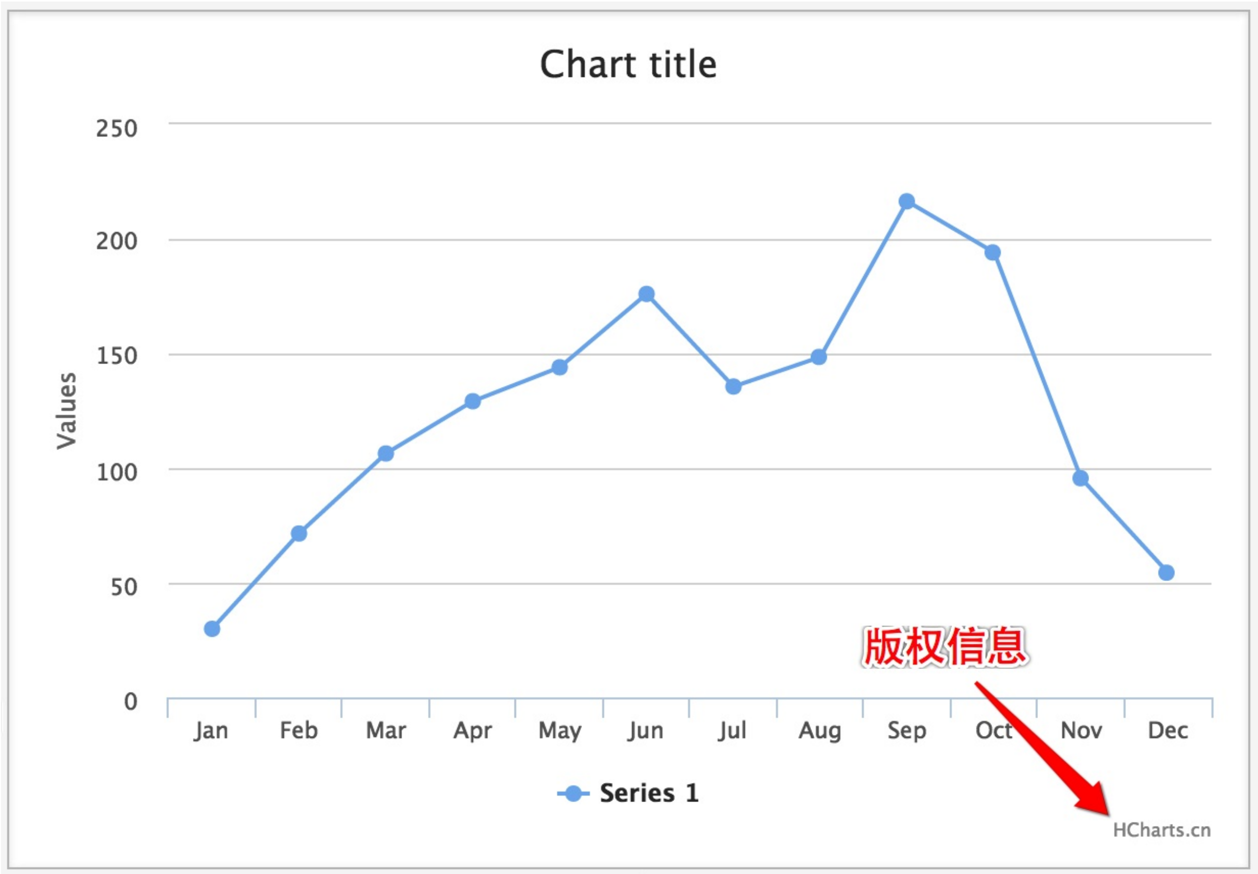
可以，同样是通过官方插件实现，见 “Vlue in legend”

更多图例相关的插件见： http://www.hcharts.cn/p/index.php?s=legend

（正文完，最后更新时间：2015-08-06 21:12:50）

版权信息（credits）

图表版权信息。显示在图表右下方的包含链接的文字，默认是文字是 Highcharts，链接是 Highcharts官网地址。通过指定 `credits.enabled=false` 即可不显示该信息。



一、版权信息的常用属性

版权信息常用的属性包括是否启用版权信息、文字、链接、位置等，详细列表如下：

属性名	描述	默认值
enabled	是否显示版权信息	true
href	版权信息点击之后指向的URL(设置自己版权信息地址时记得加“http://“)	http://www.highcharts.com
text	显示的版权信息文字	Highcharts.com
position	文字显示位置。支持的属性有align(左右对齐),verticalAlign(上下对齐), x(x轴偏移量), y(y轴偏移量)	position: { align: 'right', x: -10, verticalAlign: 'bottom', y: -5}
style	版权信息标签的CSS样式	style: { cursor: 'pointer', color: '#909090', fontSize: '10px'}

没有更多属性了，请参考 API文档：credits

二、版权信息常见设置

1、屏蔽版权信息

版权信息是默认显示的，若不需要可以通过增加代码：

```
credits:{
     enabled:false // 禁用版权信息
}
```

如果你觉得在每个图标中都配置 `credits.enabled` 很麻烦，你也可以通过修改 highcharts 源码来针对所有的图表禁用版权信息。

2、设置位置

默认右下角，**align** 可配置参数 `left` 、 `right` 、 `center` ；**verticalAlign** 可配置参数： `bottom` 、 `top` 、 `middle` ；**x** 和 **y** 分别配置的是指定位置的偏移量，负数代表向左或向上偏移，正数代表向右或者向下偏移。

3、完整的例子

```
credits:{
    // enabled:true,                    // 默认值，如果想去掉版权信息，设置为false即可
    text:'www.hcharts.cn',               // 显示的文字
    href:'http://www.hcharts.cn',    // 链接地址
    position:{                          // 位置设置
        align: 'left',
        x: 400,
        verticalAlign: 'bottom',
        y: -100
    },
    style: {                            // 样式设置
        cursor: 'pointer',
        color: 'red',
        fontSize: '30px'
    }
}
```

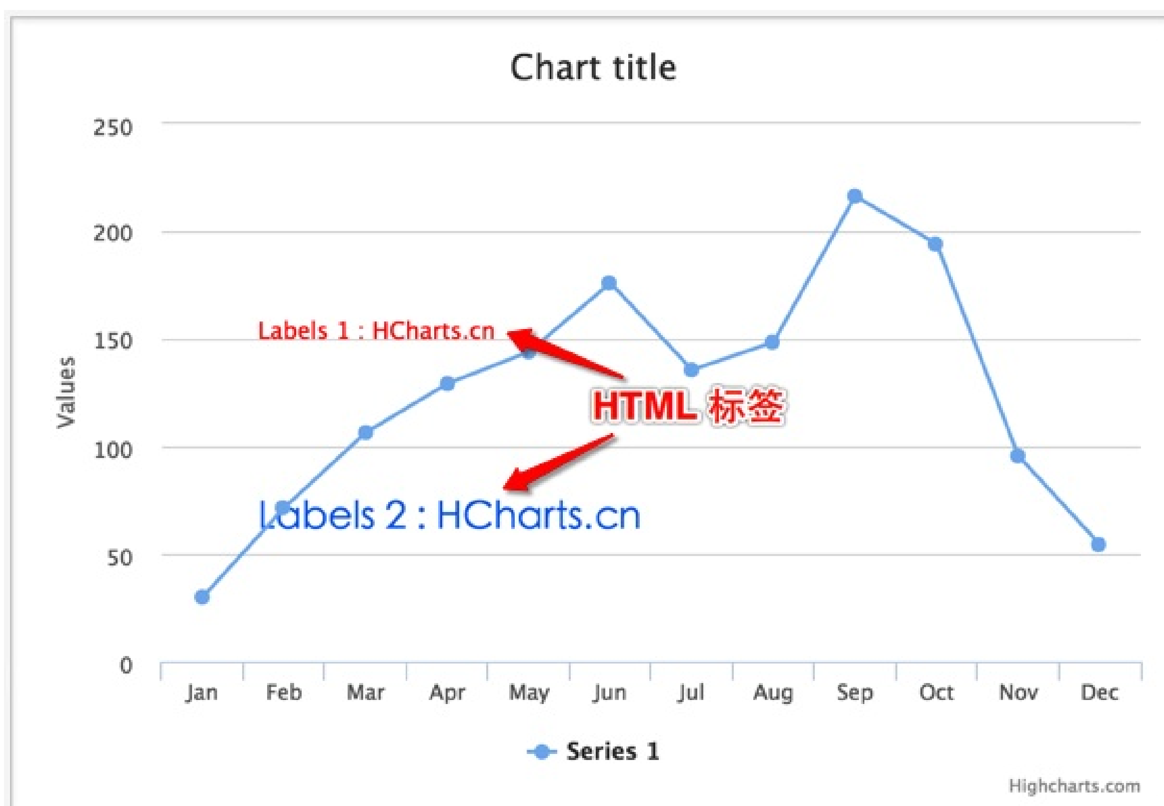
在线试一试

（正文完，最后更新时间：2015-06-10 11:08:26）

HTML标签（labels）

HTMl标签（Lables）指的是可以放置在图表中任意位置的文字标签，由于最终的文字标签是以 SVG 渲染的，所以标签的内容只支持少量的 HTML 标签，包括：<b>、<strong>、<i>、<em>、
、<span>，其中 可以通过 style 属性来设定样式，但是有效的样式仅限和文字相关的属性。HTML 标签的基本构造是：

```
labels: {
    style: {                         // 标签全局样式
        color: "#ff0000",
        fontSize: '12px',
        fontWeight: 'normal',
        fontFamily: ''
    },
    items: [{                       // items 数组，可以设置多个标签
        html: 'html 标签内容',
        style: {                    // 标签样式，会继承和重写上层全局样式
            left: '50px',
            top: '100px',
            color: 'red',
            fontSize: '12px',
            fontWeight: 'normal',
            fontFamily: ''
        }
    }]
},
```



在线试一试

扩展内容

通过学习上面的内容我们知道，HTMl标签只能添加简单的文字标签，并且只能是在图标初始化的时候才能添加，那么对于添加文字标签，highcharts 有没有更方便的编程接口呢？

答案是有的，对应的 API 是 `[Renderer](http://www.hcharts.cn/api/index.php#Renderer)` 。

Renderer 是一个提供了原始绘图接口的对象，可以直接在图表上绘制基础的图形，包括圆形、矩形、线条、文字等，在主流浏览器中，对应的是 SVG 封装，IE8 以下则是 VML 封装。

Renderer 可以通过 chart.renderer （chart 为已经存在的图表对象）或 Highcharts.Renderer() 方式调用，对应的初始化方式有所不同：

`chart.renderer`

`Highcharts.Renderer(parentNode, width, height);`

其中 parentNode 表示图形希望被添加到的 html元素（dom）。

Renderer 支持链式编程，即可以在同一个表达式中多次调用相关的函数，例如：

```
chart.render.rect(
        // ... 省略代码
    ).attr(
        // ... 省略代码
    ).css(
        // ... 省略代码
    );
```

更多关于 Renderer 的信息请参看 API 文档：Renderer。

通过 Renderer 给图表添加文字标签

1、Renderer.text()

构造方法： `Renderer.text(String str, Number x, Number y)`

参数列表：

```
String str: 需要添加的文字
Number x:   水平偏移
Number y:   竖直偏移
```

在线试一试

2、Renderer.label()

Renderer.label() 支持更多高级属性，例如边框，背景等。

构造方法：

```
Renderer.label (String str, Number x, Number y, String shape, Number anchorX, Number ancho
```

参数列表：

```
String str:       标签内容
Number x:         水平偏移
Number y:         竖直偏移
String shape:     形状
Number anchorX:   如果形状中包含指示，例如 chevron 和 callout。anchorX 指定指示形状的 x 位置
Number anchorY:   如果形状中包含指示，例如 chevron 和 callout。anchorY 指定指示形状的 y 位置
Boolean useHTML:  是否开启 HTML 模式来渲染标签
Boolean baseline:是否让标签以文字的 baseline 来竖直对齐
String className:标签的父级元素 g 的类
```



在线试一试

（正文完，最后更新时间：2015-06-15 11:50:33）

标示线（plotLines）

标示线是用来标记坐标轴上特殊值的一条直线，在绘图区内绘制一条自定义的线。标示线总是垂直于它属于的轴。它可单独定义在x轴或y轴，也可以同时定义在x轴和y轴。如果标示线同时定义在x轴和y轴，定义在y轴的标示线会显示在前面。具体实例如下：

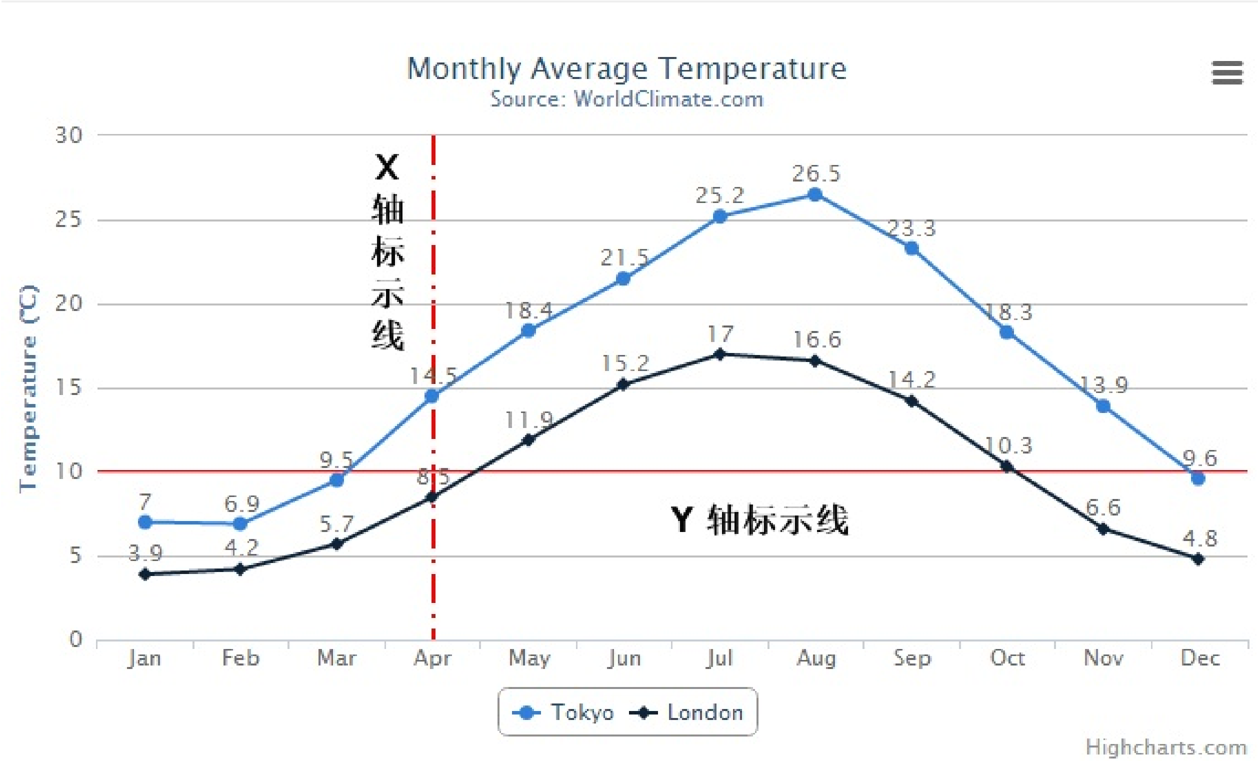
1、在x轴上值为3的地方画一条红色的宽度为2px的线

```
xAxis: {
    // ... 省略代码
    plotLines:[{
        color:'red',            //线的颜色，定义为红色
        dashStyle:'longdashdot',//标示线的样式，默认是solid（实线），这里定义为长虚线
        value:3,                //定义在哪个值上显示标示线，这里是在x轴上刻度为3的值处垂直化一条线
        width:2                 //标示线的宽度，2px
    }]
 }
```

2、在y轴画一条和x轴一样的红色的2px的线

```
yAxis: {
    // ... 省略代码
    plotLines:[{
        color:'red',           //线的颜色，定义为红色
        dashStyle:'solid',     //默认值，这里定义为实线
        value:3,               //定义在那个值上显示标示线，这里是在x轴上刻度为3的值处垂直化一条线
        width:2                //标示线的宽度，2px
    }]
}
```

上述定义的两个标示线效果如下图所示：



一、标示线的常见属性

标示线是x轴或y轴上的标记特殊刻度的线，它的属性包括了颜色，事件，id，标签，和zIndex层叠，常见属性如下表所示：

属性名	描述	默认值
color	标示线的颜色	null
dashStyle	标示线的线条样式，默认是solid，即直线型，更多下面详细说明	'solid'
events	标示线的事件，详细事件下文详解	null
id	定义标示线，在Axis.removePlotLine中定义去除那条标示线	null
value	在坐标轴上显示的位置	null
label	标示线的文字标签，用来描述标示线	null
width	标示线的宽度	null
zIndex	层叠，标示线在图表中显示的层叠级别，值越大，显示越向前，默认标示线显示在数据线之后	null

常用属性详解

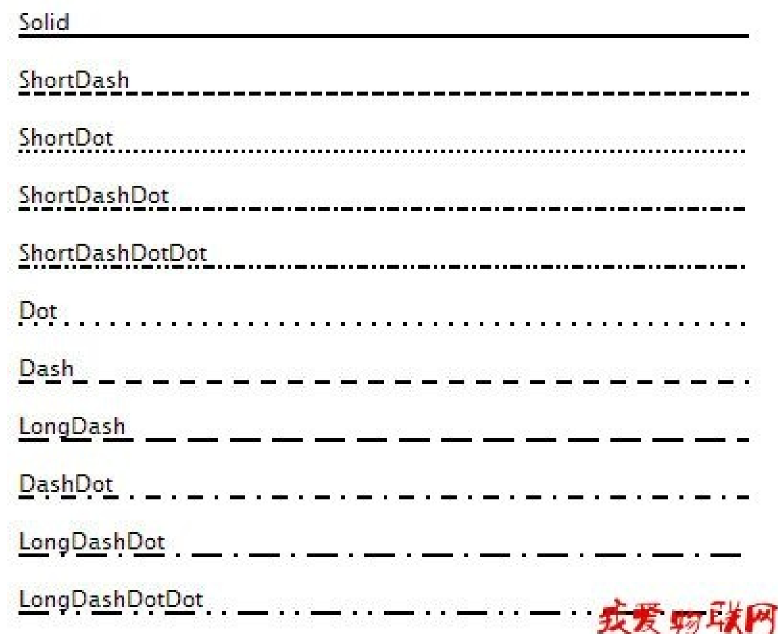
1、Labels：标签

标签是对标示线的一个文字说明，文本值默认会显示在标示线的上部。给标示线添加一个标签的实例代码如下：

```
plotLines:[{
    // ..., 省略代码
    label:{
        text:'我是标示线的标签',      //标签的内容
        align:'left',                  //标签的水平位置，水平居左,默认是水平居中center
        x:10                           //标签相对于被定位的位置水平偏移的像素，重新定位，水平居左10p
    }
}]
```

2、dashStyle ：线条样式

线条有直线型、虚线型等，所有的线条及样式如下图所示：



上图所示的线条样式同样适用于Highcharts图表中所有线条

3、Events：事件

Highcharts为标示线提供了很多相关事件，详细描述如下

```
plotLines:[{
    //..., 省略代码
    events:{
        click:function(){
             //当标示线被单击时，触发的事件
        },
        mouseover:function(){
            //当标示线被鼠标悬停时，触发的事件
        },

        mouseout:function(){
            //当标示线被鼠标移出时，触发的事件
        },

        mousemove:function(){
            //当标示线被鼠标移动(时，触发的事件
        }
    }
}]
```

更多关于标题的属性请参考API文档：xAxis plotLines 、 yAxis plotLines

二、动态增加或删除标示线

Highcharts提供了相应的函数方便在图表绘制完毕后对标示线动态的增加或删除操作。

1、增加标示线

可以通过 `addPlotLine()` 函数增加标示线，该函数的构造如下

`addPlotLine (Object options)` ，其中options是一个 `plotline` 对象，实例代码如下：

```
var chart = new Highcharts.Chart();      // Highcharts构造函数
chart.xAxis[0].addPlotLine({             //在x轴上增加
    value:2,                             //在值为2的地方
    width:2,                             //标示线的宽度为2px
    color: '#FCFFC5',                    //标示线的颜色
    id: 'plot-line-1'                    //标示线的id，在删除该标示线的时候需要该id标示
});
```

2、删除标示线

Highcharts提供函数 `removePlotLine()` 供动态删除标示线，removePlotLine ()函数结构如下：
`removePlotLine (id)`

参数说明： **id**: 标示线的id，不存在该id时，该函数式无效的

实例代码：

```
var chart = new Highcharts.Chart();             // Highcharts构造函数
chart.xAxis[0].removePlotLine('plot-line-1'); //把id为plot-line-1的标示线删除
```

通过上述的两个方法，addPlotLine()和removePlotLine(),可以动态的实现增加和删除标示线，需要注意的是，需要进行删除的标示线，在新增或初始化的时候需要给其id属性赋唯一的值，如果不存在id，removePlotLine()会失效。

在线试一试

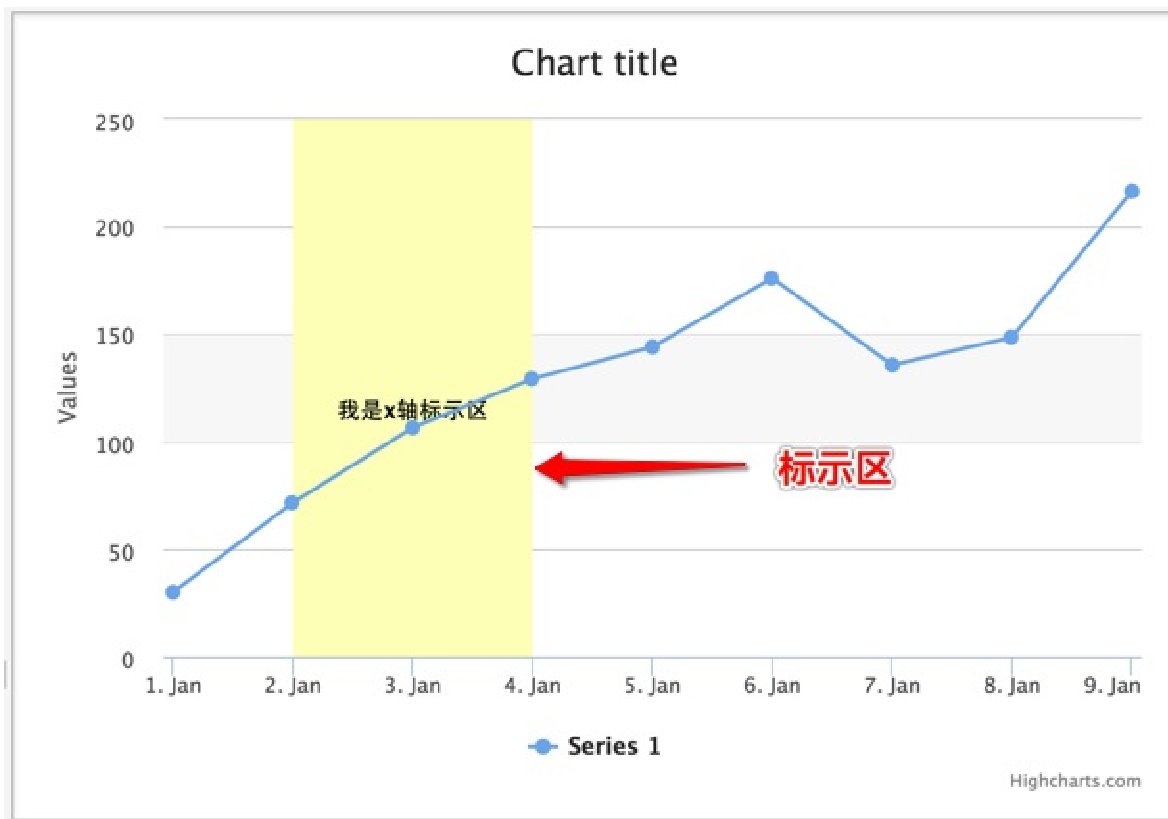
3、 在仪表图和雷达图中的标示线

在仪表图（gauge）和雷达图（polar）中，定义在x轴上的标示线将会显示一条直线，定义在y轴上的标示线将会显示一个同心圆。

（正文完，最后更新时间：2015-12-01 18:46:56）

标示区（plotBands）

标示区同标示线，只不过标示的内容为一段范围。标示区的作用、事件等很多都类似标示线，这里就只是简单说明标示区的相关内容，不做过多累述。



一、标示区的基本配置

```
xAxis: {
    // ... 省略代码
    plotBands: [{
        from: ,             // 标示区开始值
        to: ,               // 标示区结束值
        label: {            // 标示区文字标签配置，详见API
        },
        color: '',          // 标示区背景颜色
        borderWidth: ,      // 标示区边框宽度
        borderColor: ,      // 标示区边框颜色
        id: ,               // 标示区 id，用于删除等操作
        zIndex:,            // 标示区层叠，用于调整显示层次
        events: {           // 事件，支持 click、mouseover、mouseout、mousemove等事件
            click: function(e) {
            },
            mouseover: function(e) {
            },
            mouseout: function(e) {
            },
            mousemove: function(e) {
            }
        }

    }]
},
yAxis: {
    // ... 省略代码
    plotBands: [{
        // 标示区配置，同上
    }]
}
```

在线试一试

二、动态增加或删除标示区

同标示线一样，我们可以通过 Axis.addPlotBand 及 Axis.removePlotBand 来动态增加或删除标示线。

1、动态增加标示区

```
 // 实例化图表并保存图表对象
var chart = new Highcharts.Chart();

var axis = chart.xAxis[0];

axis.addPlotBand({
   id: 'myXAxisPlotBand',     // id 用于后续删除用
   from: 20,
   to: 40,
   // ...
});
```

在线试一试

2、动态删除标示区

```
var chart = new Highcharts.Chart();

var axis = chart.xAxis[0];

axis.removePlotBand('myXAxisPlotBand')
```

在线试一试

（正文完，最后更新时间：2015-09-11 09:48:22）

图表缩放（Zoom）

图表缩放指的是局部放大或缩小图表，可以更方便的查看图表数据。

Highcharts支持两种方式缩放，缩放（zoom）和平移（panning），并且是完美支持移动端手势操作的 。

一、缩放（zoom）

只需要简单设置 zoomType 即可实现图表缩放，例如：

```
$("#container").highcharts({
  chart: {
      zoomType: 'x'
  }
  // ... 省略代码
});
```

其中 zoomType 取值为 `x` 、`y` 、`xy` 中的一个，分别表示图表可以沿 x 轴，y轴，xy轴放大，也就是水平、竖直、平面放大。zoomType 默认值是 None，即无方法功能。

在线试一试

1、重置缩放比例按钮

和重置缩放比例按钮先关的配置有三个：

- resetZoomButton：按钮相关配置，详见API文档
- lang.resetZoom：按钮文字
- lang.resetZoomTitle：按钮标题

在线试一试

2、选中样式

selectionMarkerFill 为选中时区域的背景填充，指为颜色（支持颜色代码、十六进制、rgb、rgba形式）.

在线试一试

3、事件

图表缩放事件处理函数，chart.events.selection，在事件处理函数里，可以获取缩放相关信息，例如缩放后图表的范围，示例代码

```
$("#container").highcharts({
  chart: {
    events: {
      selection: function(e) {
        // 事件处理代码，可以通过 console.log(e) 查看更多详细信息
      }
    }
  }
});
```

缩放事件的一些应用，例如需要在新的图表里展现当前选中的范围曲线，而不是放大操作，这时候就需要用到这个事件函数了，相关说明见中文社区帖子。

在线试一试

二、平移（panning）

图表缩放后，我们还可以进行平移操作（Highstock 默认就是平移操作）。默认情况下，highcharts 是没有开启平移功能的，这个下面几个参数相关

- panning：是否开启平移功能，highcharts 默认是 false，highstock 默认为 true
- panKey：平移按键，对应的是键盘的键名，例如 'Shift'，'ctrl'。对于 highcharts，开启平移后，还需要设置 pankey，对应的操作是缩放图表后，按钮指定按键就可以平移了；对于 Highstock 则没有这个配置，默认平移是直接拖动操作的。
- pinchType：同 zoomType，用于移动端手势操作缩放方向。

在线试一试、在线试一试2

（正文完，最后更新时间：2015-10-26 09:29:35）

语言文字（lang）

图表中的文字及语言相关的内容都是可以自定义及本地化的，下面详细说明。

一、语言及文字

Highcharts 中的文字可以通过 Highcharts.setOptions.lang 来设定，lang 属于全局配置，对当前页面的所有图表有效，对应的汉化后的配置是：

```
// 全局配置，对当前页面的所有图表有效

Highcharts.setOptions({
    lang:{
       contextButtonTitle:"图表导出菜单",
       decimalPoint:".",
       downloadJPEG:"下载JPEG图片",
       downloadPDF:"下载PDF文件",
       downloadPNG:"下载PNG文件",
       downloadSVG:"下载SVG文件",
       drillUpText:"返回 {series.name}",
       loading:"加载中",
       months:["一月","二月","三月","四月","五月","六月","七月","八月","九月","十月","十一月","十
       noData:"没有数据",
       numericSymbols: [ "千" , "兆" , "G" , "T" , "P" , "E"],
       printChart:"打印图表",
       resetZoom:"恢复缩放",
       resetZoomTitle:"恢复图表",
       shortMonths: [ "Jan" , "Feb" , "Mar" , "Apr" , "May" , "Jun" , "Jul" , "Aug" , "Se
       thousandsSep:",",
       weekdays: ["星期一", "星期二", "星期三", "星期四", "星期五", "星期六","星期天"]
    }
});
```

在线试一试

二、时间及时区

1、时间格式化

Highcharts 不同位置的时间显示可以通过对应时间格式化函数、Highcharts.dateFormate来处理，例如处理数据提示框中的时间显示可以通过下面的方法实现：

```
tooltip: {
    dateTimeLabelFormats: {
        year:"%Y",
        second:"%Y-%m-%d %H:%M:%S",
        // ...
    },
    // 还可以在 格式化函数中通过调用  Highcharts.dateFormate() 函数来处理，总之是非常灵活的
}
```

对于 x轴的时间格式化可以通过下面的方式实现：

```
xAxis: {
    dateTimeLabelFormats: {
        year: '%Y',
        month: '%Y-%m',
        dat: '%Y-%m-%d',
        // ...
        // 当然还可以通过xAxis.labels.formatter 函数来格式化
    }
}
```

在线试一试

对于其它地方需要进行日期格式化这里不再累述。

2、时区

通过 global.timezoneOffset 可以设置时区，中国属于 +8 区，所以有的时候图表中显示的时间和实际时间相差 8 个小时，这时候我们可以通过设置时区来修正，配置代码如下：

```
Highcharts.setOptions({
    global: {
        timezoneOffset: -8 * 60  // +8 时区修正方法
    }
})
```

三、符号

图表中数值显示时往往带有格式化符号，这里统一说明如下：

1、小数点、千分号、公制前缀

```
Highcharts.setOptions({
    decimalPoint: '.',    // 小数点号，例如 12.50
    thousandsSep: ','     // 千分号，例如 12,000
    numericSymbols: 'k,M,G,T,P,E' // 公制前缀，通过设置为 null 不显示 12k，1.2M 这种形式
})
```

2、数值格式化函数

通过 Highcharts.numberFormat 可以对图表中的数值进行格式化，函数说明如下

函数构造

Highcharts.numberFormat (Number number, [Number decimals], [String decimalPoint], [String thousandsSep])

参数说明：

- number 需要格式化的数值

- decimals 精度，保留位置，可选参数，默认是 0
- decimalPoint 小数符号，可选参数，默认是 global.decimalPoint
- thousandsSep 千分符，可选参数，默认是 global.numericSymbols

点击查看详情

（正文完，最后更新时间：2015-12-01 18:44:51）

标签及字符串格式化（Format）

（正文完，最后更新时间：）